

LIBRARY
VALLEY SCHOOL
MONTANA 59711, CALIFORNIA 95945-8002

NAVAL POSTGRADUATE SCHOOL

Monterey, California



THESIS

T82317

ADAPTIVE GO-BACK-N
AN ARQ PROTOCOL FOR
A TACTICAL VSAT NETWORK

by

David T. Tsuda

September 1989

Thesis Advisor:

Tri T. Ha

Approved for public release; distribution is unlimited

REPORT DOCUMENTATION PAGE

1a Report Security Classification Unclassified			1b Restrictive Markings		
2a Security Classification Authority			3 Distribution Availability of Report		
2b Declassification Downgrading Schedule			Approved for public release; distribution is unlimited.		
4 Performing Organization Report Number(s)			5 Monitoring Organization Report Number(s)		
6a Name of Performing Organization Naval Postgraduate School		6b Office Symbol (if applicable) 32	7a Name of Monitoring Organization Naval Postgraduate School		
6c Address (city, state, and ZIP code) Monterey, CA 93943-5000			7b Address (city, state, and ZIP code) Monterey, CA 93943-5000		
8a Name of Funding Sponsoring Organization		8b Office Symbol (if applicable)	9 Procurement Instrument Identification Number		
8c Address (city, state, and ZIP code)			10 Source of Funding Numbers		
			Program Element No	Project No	Task No
			Work Unit Accession No		
11 Title (include security classification) ADAPTIVE GO-BACK-N AN ARQ PROTOCOL FOR A TACTICAL VSAT NETWORK					
12 Personal Author(s) David T. Tsuda					
13a Type of Report Master's Thesis		13b Time Covered From To		14 Date of Report (year, month, day) September 1989	
15 Page Count 84					
16 Supplementary Notation The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.					
17 Cosati Codes			18 Subject Terms (continue on reverse if necessary and identify by block number)		
Field	Group	Subgroup	VSAT, satellite communication system, adaptive protocol, ARQ.		
19 Abstract (continue on reverse if necessary and identify by block number) Tactical satellite networks require a Data Link Control (DLC) layer protocol to provide a virtual error-free communications link for point-to-point communications. It is of particular interest that the DLC protocol continues to operate under high bit error ratio conditions due to added noise caused by interference from other communications systems or intentional jamming. Automatic Repeat Request (ARQ) protocols are the most commonly used DLC protocols in commercial systems. However, under high bit error ratio conditions, the throughput efficiency of an ARQ protocol decreases rapidly. To improve the throughput efficiency of ARQ protocols, some adaptive ARQ strategies have been theoretically analyzed. One particular ARQ protocol, an adaptive Go-Back-N (GBN) protocol, was selected for implementation in a tactical satellite network. The throughput efficiency of the adaptive GBN protocol was evaluated using data produced by a computer simulation. The simulation results for a three-stage adaptive GBN protocol revealed a severe decrease in throughput efficiency when the bit error ratio was sufficient to cause frequent transitioning between the second and third stages. Under increasing bit error ratio conditions, a simulation of a two-stage adaptive GBN protocol demonstrated an appreciable improvement in throughput efficiency over a standard GBN protocol and the three-stage adaptive GBN protocol.					
20 Distribution Availability of Abstract <input checked="" type="checkbox"/> unclassified unlimited <input type="checkbox"/> same as report <input type="checkbox"/> DTIC users			21 Abstract Security Classification Unclassified		
22a Name of Responsible Individual Tri T. Ha			22b Telephone (include Area code) (408) 646-2788		22c Office Symbol 6211a

T245485

Approved for public release; distribution is unlimited.

Adaptive Go-Back-N
An ARQ Protocol for
a Tactical VSAT Network

by

David T. Tsuda
Captain, United States Army
B.S.E.E., California Polytechnic State University, 1979.

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
September 1989

ABSTRACT

Tactical satellite networks require a Data Link Control (DLC) layer protocol to provide a virtual error-free communications link for point-to-point communications. It is of particular interest that the DLC protocol continues to operate under high bit error ratio conditions due to added noise caused by interference from other communications systems or intentional jamming. Automatic Repeat Request (ARQ) protocols are the most commonly used DLC protocols in commercial systems. However, under high bit error ratio conditions, the throughput efficiency of an ARQ protocol decreases rapidly. To improve the throughput efficiency of ARQ protocols, some adaptive ARQ strategies have been theoretically analyzed. One particular ARQ protocol, an adaptive Go-Back-N (GBN) protocol, was selected for implementation in a tactical satellite network. The throughput efficiency of the adaptive GBN protocol was evaluated using data produced by a computer simulation. The simulation results for a three-stage adaptive GBN protocol revealed a severe decrease in throughput efficiency when the bit error ratio was sufficient to cause frequent transitioning between the second and third stages. Under increasing bit error ratio conditions, a simulation of a two-stage adaptive GBN protocol demonstrated an appreciable improvement in throughput efficiency over a standard GBN protocol and the three-stage adaptive GBN protocol.

11.5.013
82317
C.L.

THESIS DISCLAIMER

The reader is cautioned that computer programs developed in this research may not have been exercised for all cases of interest. While every effort has been made, within the time available, to ensure that the programs are free of computational and logic errors, they cannot be considered validated. Any application of these programs without additional verification is at the risk of the user.

A. INTRODUCTION	30
B. SIMULATION DEVELOPMENT	30
1. Model Parameters and Assumptions	30
2. Simulation Program	31
C. SIMULATION RESULTS	32
D. CONCLUSION.	36
APPENDIX A. SIMULATION INPUT FILES	37
APPENDIX B. SIMULATION FLOW DIAGRAM	61
LIST OF REFERENCES	73
INITIAL DISTRIBUTION LIST	75

LIST OF FIGURES

Figure 1. Star configuration	4
Figure 2. Full-mesh configuration	5
Figure 3. Seven layer OSI Network Architecture	11
Figure 4. Information frame format	13
Figure 5. Supervisory or Unnumbered frame format	14
Figure 6. Control field format	15
Figure 7. GBN Protocol: (a) handling of ACK/NAK; (b) timeout delay	19
Figure 8. Adaptive GBN: One-way Information Transfer	25
Figure 9. Adaptive GBN: Two-way Information Transfer	26
Figure 10. Throughput Efficiency for GBN vs AGBN (3 stages)	34
Figure 11. Throughput Efficiency for GBN vs AGBN (2 stages)	35

I. INTRODUCTION

The purpose of this thesis was to investigate the throughput performance of an adaptive Automatic Repeat reQuest (ARQ) protocol for use with a tactical satellite communications network, specifically a very small aperture terminal (VSAT) network.

Satellite networks offer the advantage of providing communications to users scattered over a large geographical area with high reliability. A VSAT network offers the advantage of networking a large number of interactive remote terminals and providing processing gateways to interface other networks. As with any tactical communications network, a VSAT network is susceptible to interference from other communications systems and intentional jamming. A properly designed tactical satellite network can minimize interference and reduce probability of detection. However, a high bit error ratio (BER) is still expected due to jamming (i.e., increased jamming-to-signal ratio (JSR)).

In a tactical communications network subjected to interference, intentional or not, information must continue to be transferred reliably. An ARQ protocol is a Data Link Control (DLC) level protocol whose purpose is to provide an error-free virtual communications link between two stations. It is already known that, as the bit error ratio of a transmission path increases, the throughput performance of an ARQ protocol decreases rapidly. In this thesis, the strategy of an adaptive ARQ protocol, specifically, adaptive Go-Back-N (GBN), is discussed. A simulation of adaptive GBN between two VSAT stations is performed at increasing bit error ratios (BER) and the adaptive GBN protocol's throughput performance is compared to the standard GBN protocol throughput performance.

The selection of a Data Link Control (DLC) protocol for a tactical VSAT network is dependent on the network architecture (i.e., network configuration, modulation scheme, traffic handling capabilities, network control protocol, etc.). Minimum use of frequencies, maximizing information throughput, and maximizing system reliability and availability are some of the key design goals and restrictions used in formulating the tactical VSAT network. Chapter 2 presents an overview of a tactical VSAT network architecture. Chapter 3 presents the functions of a DLC protocol, its structure, and data transfer modes. Chapter 4 discusses ARQ protocols. Chapter 5 presents the adaptive

GBN protocol strategy and implementation. Chapter 6 presents the development of the simulation and results.

II. TACTICAL VSAT NETWORK ARCHITECTURE - AN OVERVIEW

A. GENERAL

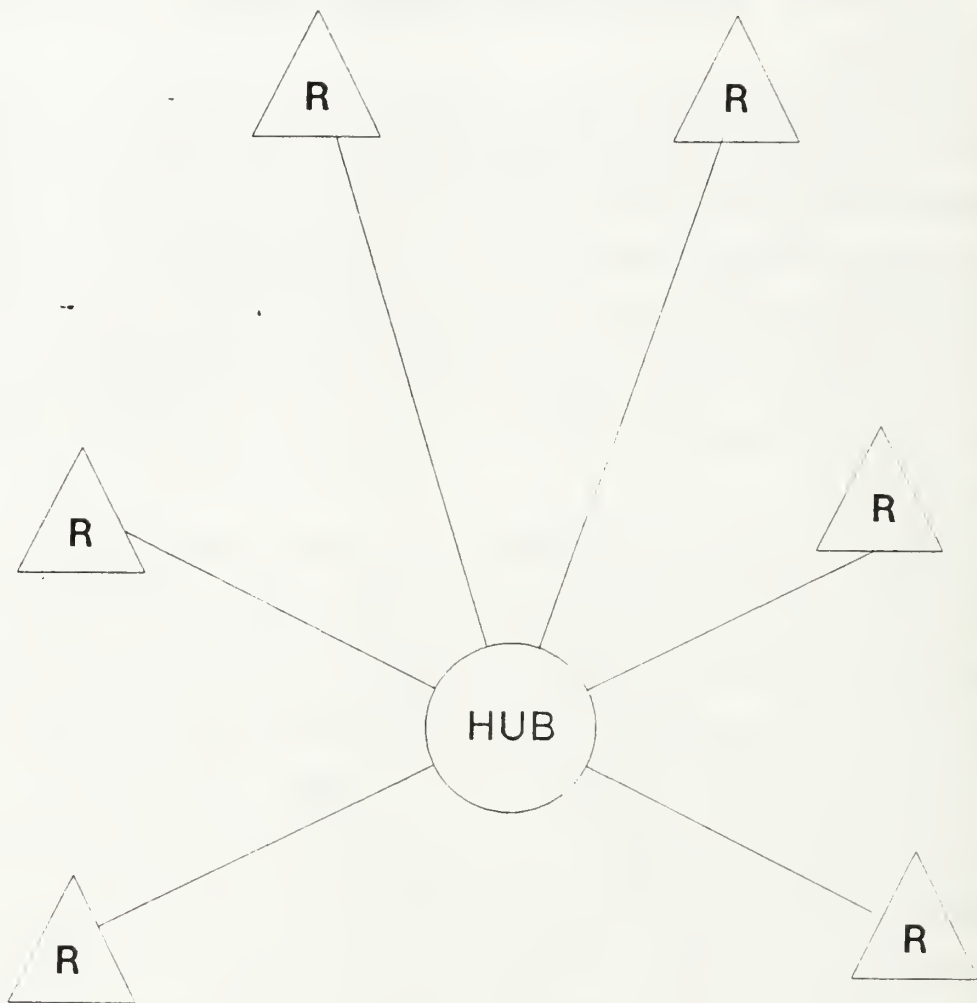
A VSAT satellite communications network is a network of low-cost satellite terminals with small antenna dishes (typically less than two meters in diameter), solid-state transmitters with low transmission power (typically 2 watts or less), low-cost low noise amplifiers and convertors, and low data rates (9.6 kbps to 128 kbps). The range of operating frequencies is 8-12 GHz for military use and 11-14 GHz for commercial use. To transmit at higher data rates, the VSAT station must increase transmission power or antenna size, or both. The technology for VSAT networks is a mature technology. Currently, several VSAT networks are in operation in the U.S. serving a wide variety of customers. [Ref. 1]

Current tactical satellite communication systems require wide bandwidths to support high data rates. The author's experience with tactical satellite communications in foreign countries has shown that availability of authorized satellite frequencies with wide transmission bandwidths is very limited. Therefore, only a few tactical satellite communications systems are deployed to support a multitude of users. Generally, several users must be connected by terrestrial lines to a single satellite terminal. Like an astronaut tethered to a spacecraft, the users have limited deployment capabilities without losing communication support. As a tactical communications system, the remote terminals of a VSAT network are lightweight so that it will be faster to deploy, install, operate and redeploy terminals as the tactical situation changes. A tactical VSAT network is better suited to provide communications support to a larger number of users spread over a wide geographic region. The tactical VSAT network be may deployed as initial communications systems to regions void of commercial facilities or to restore immediately critical communications links that have failed.

B. NETWORK CONFIGURATION

There are three types of network configurations for a VSAT network. They are described as star, full-mesh and a hybrid of star and full-mesh. The selection of the configuration depends on application requirements. [Ref. 2]

Currently, most commercial applications use the star configuration as shown in Figure 1. All communications are between remote VSAT terminals and a hub terminal. The hub terminal is a medium size terminal (i.e., larger antenna of 5-7 meters in diameter

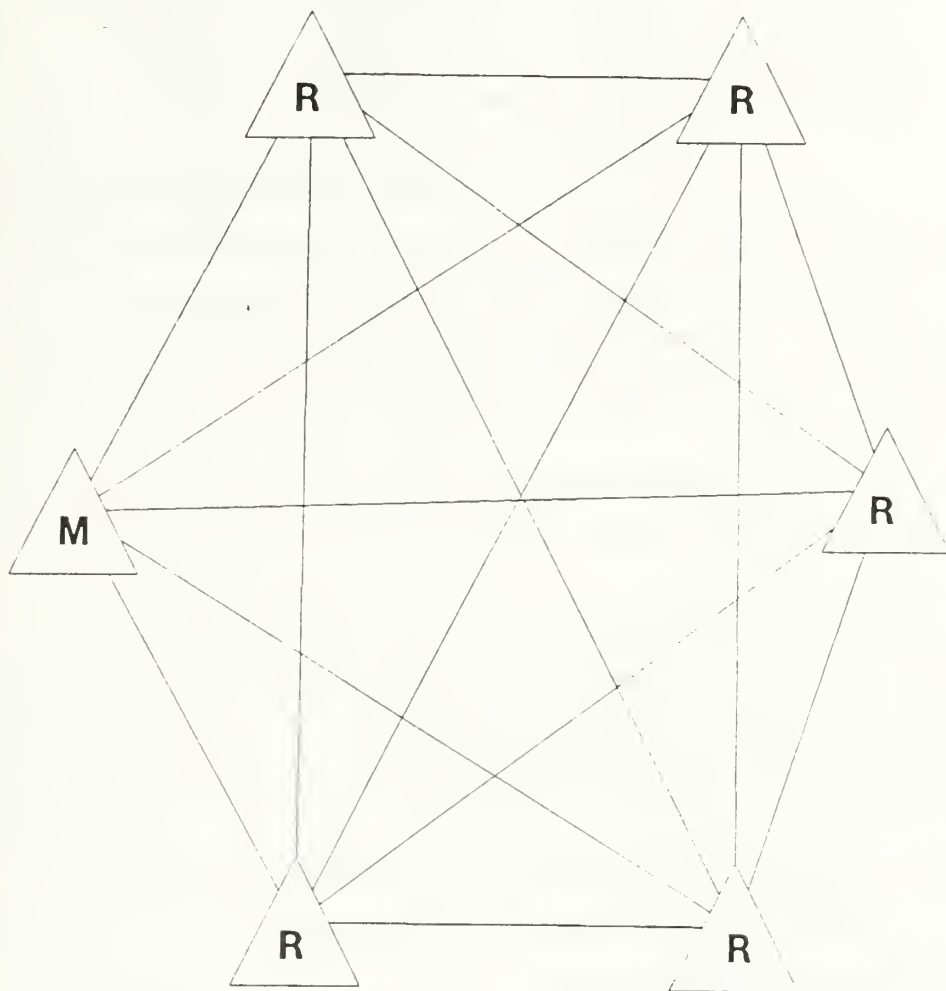


R: remote VSAT station

HUB: hub station

Figure 1. Star configuration

and higher transmission power). Single-hop transmission is used primarily when VSAT terminals need only to communicate with the hub terminal, or vice versa. The hub terminal may be a central processing point or a gateway to another network. Double-hop



R: remote VSAT station
M: master control station

Figure 2. Full-mesh configuration

communication is used when there is a need for remote VSAT terminals to transmit to other remote VSAT terminals. The information is passed to the hub terminal, processed, and then passed from the hub terminal to the destination VSAT terminal.

Double-hop communication is used when satellite delay is not a critical operational parameter. The satellite transmission delay time for a single-hop is typically 250 milliseconds. For a double-hop, the transmission delay time is 500 milliseconds plus processing time at the hub terminal.

In a full-mesh configuration as shown in Figure 2, any VSAT terminal can communicate directly with any other terminal without having to go through a hub terminal (single-hop communication). In place of a hub terminal, a master station is designated to control network activity. Any of the VSAT terminals may be a gateway to other networks. Techniques for implementing a full-mesh configuration will be discussed in the next section. The hybrid configuration is a combination of the star and full-mesh configurations. This type of configuration may be implemented when there is a mix of VSAT network requirements. There may be a requirement for some terminals to communicate with a hub terminal or master station only. Additionally, the capability exists for other terminals to communicate directly with other terminals, or indirectly via the hub terminal.

The selection of a configuration for a tactical VSAT network requires minimum transmission delay and survivability. The star configuration is not efficient when considering the added delay due to double-hop communications. It is also considered vulnerable because the hub terminal is the key to all communication links. For these reasons, a full-mesh configuration is selected to be developed. Transmission delay is limited to a single hop and more than one terminal with the capabilities of a master station can be added into the configuration for survivability.

C. MULTIPLE ACCESS SCHEMES

A VSAT network is inherently power limited due to the small antenna dishes and low transmission power. Phase shift keying (PSK), quaternary phase shift keying (QPSK), and multi-frequency shift keying (MFSK) are commonly used as modulation techniques. There are several multiple access methods in which a full-mesh configuration can be implemented. The popular multiple access schemes are time division multiple access (TDMA), frequency division multiple access (FDMA), and spread spectrum multiple access (SSMA). A VSAT network may be implemented with any combination of the three multiple access schemes. [Ref. 3] The major disadvantages of TDMA techniques are the critical system timing requirements for all terminals and the relative difficulty in adding new VSAT terminals to the network. Frequency division multiple access is implemented with each VSAT transmitting a single channel per carrier (SCPC).

The implementation of a full-mesh configuration with FDMA will require several frequency synthesizers for each terminal. The system constraints will be frequency instability and phase noise. Also, the design goal of minimizing the number of required frequencies for the network eliminates the use of FDMA. Spread spectrum multiple access schemes can operate asynchronously and do not require coordination of operating frequencies.

1. Spread Spectrum Multiple Access

The large beamwidth due to the use of a small satellite antenna dish results in significant mutual interference with adjacent satellite systems. Implementation of spread spectrum modulation techniques minimize the mutual interference effects. Spread spectrum techniques transmit a data-modulated signal such that its transmission bandwidth is much greater than the minimum bandwidth necessary to transmit the data-modulated signal. This results in a transmitted signal whose spectral density appears as low level noise over a very wide bandwidth. [Ref. 4: p. 328.] The low spectral density results in minimum interference to adjacent satellite systems and makes the signal difficult to detect. Correct demodulation results in the despreading of the data-modulated signal. When other signals in the spread spectrum transmission bandwidth (interference and jamming signals) are demodulated, they are spread across the transmission bandwidth and appear as low level noise. This rejection of interference is called the *processing gain* of the method.

The two basic spread spectrum techniques are frequency-hopping (FH) and direct sequence (DS). The FH transmitted spectrum appears as a data-modulated carrier which is randomly hopping between frequencies. The randomness is actually sequential and periodic, but extremely difficult to detect and determine. Similar to the FDMA system, frequency instability and phase noise become system constraints. Highly stable wideband frequency synthesizers are very expensive.

Direct sequence spread spectrum produces a wideband spectrum by modulating a data-modulated signal with a very wideband spreading signal. The wideband spreading signal is known as a pseudo-noise (PN) spreading code. Direct sequence spread spectrum can be implemented with code division multiple access (CDMA). With DS-CDMA, each VSAT has its own particular PN code which acts as an address. It is important to note that the crosscorrelation properties of the PN codes must be minimal. This allows all VSAT's to transmit in the same bandwidth and overlap in time. A VSAT will be able to correctly demodulate only the signal that was spread using its PN code.

All other VSAT signals and other interference signals appear as low level noise after demodulation. [Ref. 3: p. 12]

An indepth analysis of the crosscorrelation of Gold codes, a special selection of PN codes, was performed by David Hayes [Ref. 5]. The results of the analysis are included here.

“A DS-CDMA system has a receiver correlation loss resulting from crosscorrelation interference within the set of codes. This loss quantifies the receiver’s difficulty in distinguishing one code from another. Whenever multiple signals share a satellite transponder, an additional loss results from suppression of smaller signals by larger signals. [Ref. 5: p. 2] Longer codes produced lower coding interference losses, which resulted in higher signal-to-noise ratios (SNR). The use of longer codes also allows a larger number of simultaneous users. The optimal code length for a fifty terminals (i.e., fifty codes) was 1023.” [Ref. 5: p. 34]

2. Overlay Service

Overlay service is a method of adding a signal to a transponder bandwidth already considered fully occupied. In short, the spectrum of the signal to be added is overlaid within the bandwidth of the existing signal. [Ref. 6: p. 4] The DS spread spectrum signal is best suited for this type of frequency reuse service. It has already been discussed that the DS spread spectrum signal is seen as a low flat power spectrum by other receivers and the other transmitters’ signals are demodulated as low level noise by the DS-CDMA receiver. Implementation of overlay service allows the use of a direct sequence spread spectrum signal in case of emergencies.

D. NETWORK CONTROL

Many commercial VSAT systems use a collision-based protocol (e.g., ALOHA). However, it is envisioned that the tactical VSAT network will be operated under increasing bit error ratio conditions. During periods of high bit error ratios, the performance of collision-based protocols becomes unacceptable. As the bit error ratio increases, the collision-based protocols experience large processing delays, which lead to backing up of data transmissions and instability to the point where no information is being passed.

A form of Demand Assignment/Multiple Access (DAMA) is recommended for the tactical VSAT network. The master station schedules which VSATs may transmit, and how many VSATs may transmit at one time. Processing delays would be deterministic; therefore, instabilities are avoided. Master station control with DAMA of the VSAT network is essential to avoiding instabilities.

After the master station has assigned a full-duplex channel to two stations and granted them permission to communicate, a data transfer operation must be defined to ensure a virtual error-free communications link.

III. THE DATA LINK CONTROL LAYER AND DATA TRANSFER MODES

A. DATA LINK CONTROL LAYER

The International Standards Organization (ISO) developed the Open System Interconnection (OSI) network architecture to standardize the concept of a layered architecture for data networks. The OSI model consists of a hierarchy of seven functional layers as shown in Figure 3. Each layer performs specific functions for the layer above. The actual operation of transferring data between stations is performed by the first three layers of the OSI model: the physical layer, data link control layer (DLC), and the network layer, respectively.

The physical layer provides a virtual link for transmitting a sequence of bits on a physical communications channel between two stations. The physical layer consists of modules called digital data modems (modulators and demodulators) at each station which are the physical interfaces to the communications channel. The function of the modem is to translate the data bits in the form of frames from the DLC layer into signals appropriate for the communications channel, and conversely, to translate signals received from the communications channel to data bits for the DLC layer. [Ref. 7: pp. 17-19]

The physical communications channel is subject to interference from outside sources which results in noise being added to the signal. The added noise may corrupt the signal enough to result in a received sequence of bits different from the original transmitted sequence. The function of the DLC layer is to provide a virtual error-free link between two stations by a combination of error detection and frame retransmission requests. The DLC layer receives a packet of information from the network layer and adds overhead control bits to the beginning and end of a packet to form a frame. The frame is passed to the physical layer for transmission. The overhead bits are used to form the frame header and trailer as shown in Figure 4. A more detailed discussion of the frame structure is presented in the next section. One of the functions of the overhead bits is to identify if the received frame contains errors. If the received frame contains errors, the frame is rejected and a request is made to the transmitting station's DLC layer to retransmit that frame. Otherwise, the packet is passed to the network layer. [Ref. 7: pp. 20-21]

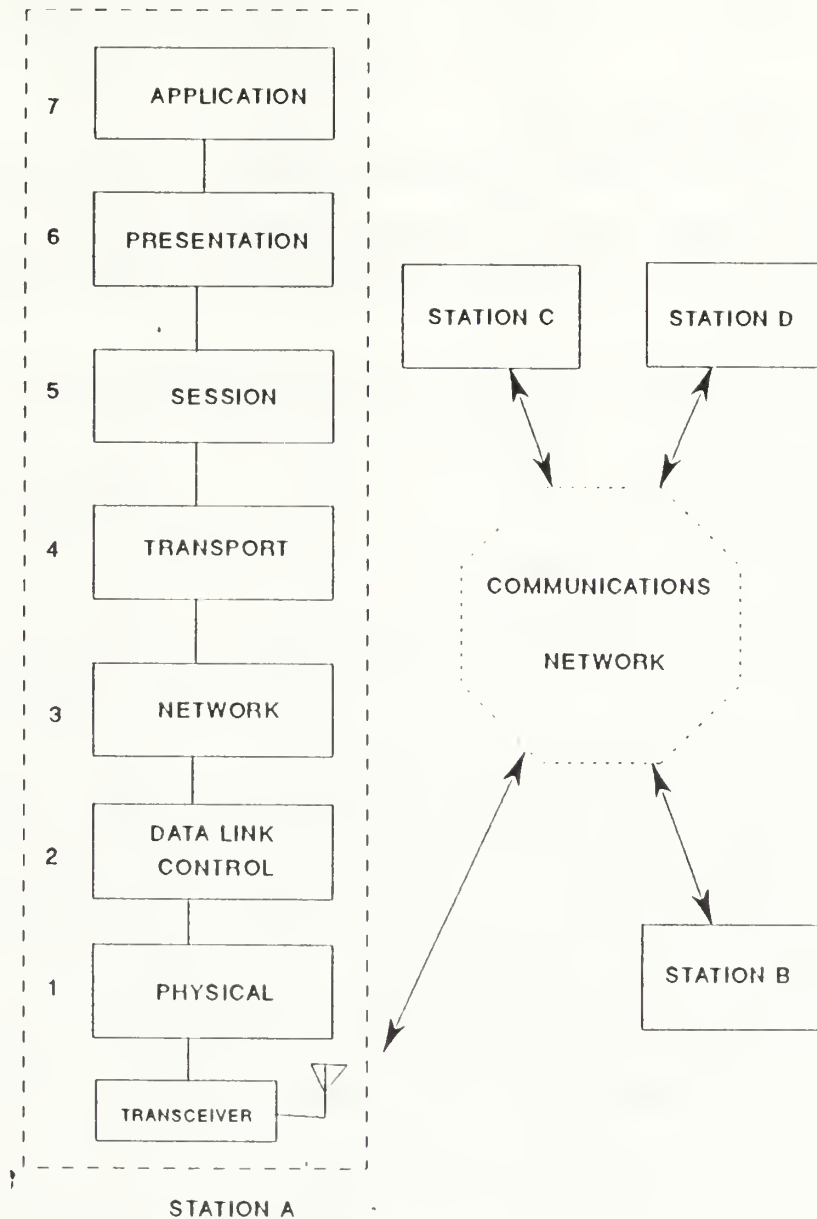


Figure 3. Seven layer OSI Network Architecture: [Ref. 10]

The network layer determines where the arriving packets are to be sent. Also, the network layer must decide when to accept packets from the higher layer and when to

transmit those packets to other stations. This function is used to exercise flow control or control traffic congestion in the network. Since there are delays in the DLC layer in accepting packets, the network layer provides buffer space for packets. [Ref. 7: pp. 22-24]

B. DLC FRAME STRUCTURE

A DLC protocol will have a specific frame structure to support data transfer procedures in establishing a virtual error-free link. There are three types of frames to perform the data transfer and link control: supervisory frame, unnumbered frame and the information frame.

The supervisory frame or S frame is used to control data transfer and contain frame acknowledgements (ACK) or negative acknowledgements (NAK). The unnumbered frame or U frame is used to carry control information necessary to initiate and terminate a data transfer session. The S and U frames do not carry any data packet information. The information frame or I frame is used to transfer the data packets received from the network layer. There are two basic formats to support the frames. Both types of formats contain five common fields: two flag fields, address field, control field and a frame check sequence (FCS) field. The first format is used for the S and U frames as shown in Figure 5. The second format is used for the I frame. The I frame contains an additional information field which may have a variable length as shown in Figure 4.

The flag fields are identical and are used to identify the beginning and the end of a frame. The beginning flag field is also used to acquire frame synchronization. The flag field is an eight bit sequence starting with an '0' bit, followed by six '1' bits and a final '0' bit. To ensure that the bit sequence between the flag fields is not duplicated (i.e., identifying a false flag field), bit stuffing is used to insert a '0' bit after any sequence of five '1' bits between the flag fields. At the receiving station, the '0' bit is removed after any sequence of five '1' bits between the two flags.

The address field is usually eight bits in length and is used to identify the intended receiver. For network with more than 256 addresses, the address field may be extended as necessary. If a network has a multi-point configuration, the address field is used to identify the intended receiving station. If a network has a point-to-point configuration, the address field specifies that the frame contains a command or a response. If the frame is intended for a receiving station, the frame contains a command.

The control field is an eight bit sequence that specifies the type and function of the frame. See Figure 6 for the control field format. An I frame is identified by a '0' bit in

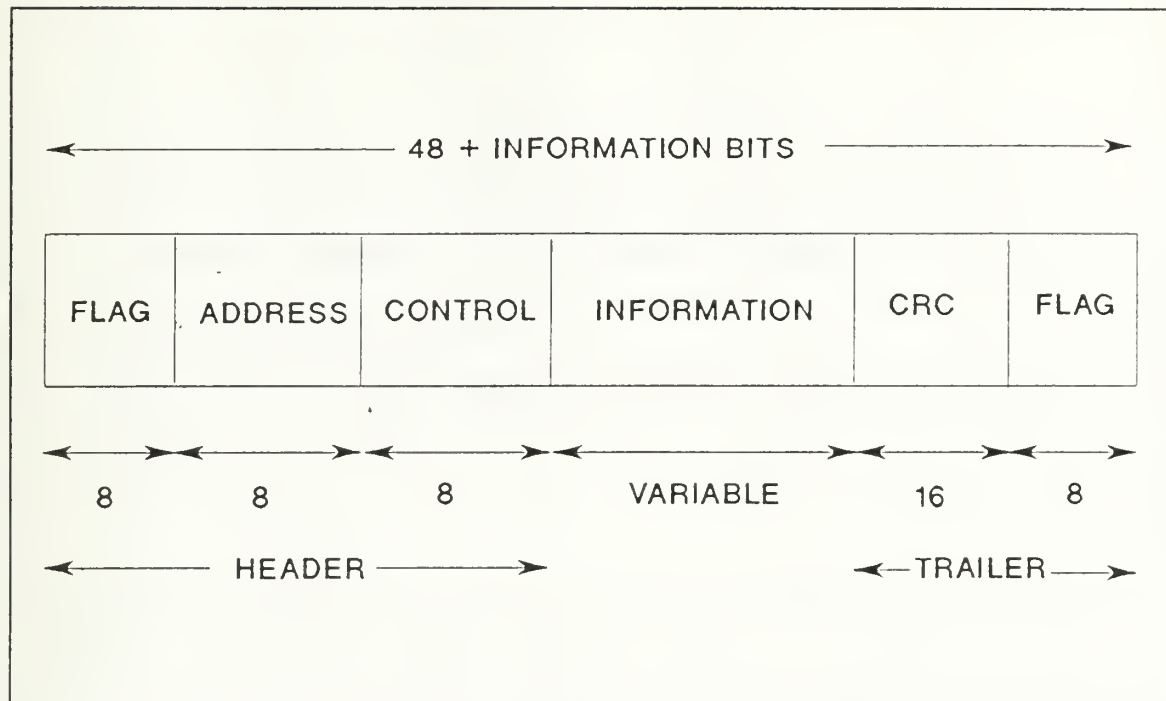


Figure 4. Information frame format

the first bit position of the control field. The next three positions specify the send sequence number SN. The Poll/Final (P/F) bit is the fifth bit position for all three frames. If the frame contains a command, the P/F bit is a P bit. If the frame contains a response, the P/F bit is a F bit. When a station sends a command, the P bit is set. The receiving station in turn sends a response and sets the F bit. The final three bit positions of an I or an S frame specify the receive sequence number RN. The RN is the sequence number of the frame the receiving station is expecting. Either format may be used to acknowledge received I frames.

The two S bits of the S frame are used to specify up to four supervisory functions. The supervisory functions are as follows:

Receive Ready (RR) Command/Response: RR frames are used to indicate readiness to receive I frames and to acknowledge I frames received.

Receive Not Ready (RNR) Command/Response: RNR frames are used to indicate a busy condition and to acknowledge I frames received.

Reject (REJ) Command/Response: REJ frames are used to request retransmission of all I frames starting from the specified RN and to acknowledge I frames received.

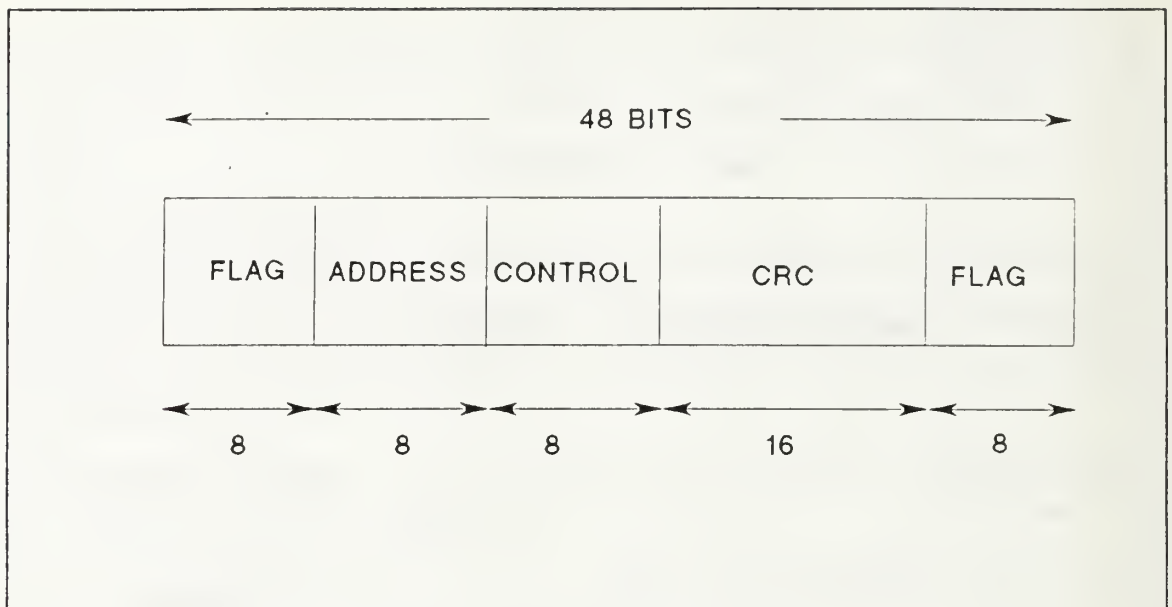


Figure 5. Supervisory or Unnumbered frame format

Selective Reject (SREJ) Command Response: SREJ frames are used to specify the number of a frame that was received in error and to request retransmission of that frame.

The five M bits of the U frame are used to specify up to 32 commands and associated 32 responses. A detailed listing of the 64 U frame commands and responses can be found in [Ref. 8: p. 122-123]. A few of the commands and responses are listed here:

Extended Numbering Set Mode (SXXME) Commands: These commands are used to establish a particular modulo 128 sequence numbering mode.

SNRME - Set Normal Response Mode Extended

SARME - Set Asynchronous Response Mode Extended

SABME - Set Asynchronous Balanced Mode Extended

Set Initialization Mode (SIM) Command: This command is used to establish the initialization mode.

Disconnect (DISC) Command: This command is used to disconnect a previously established mode and to assume the disconnected mode.

Unnumbered Poll (UP) Command: This command is used to solicit response frames from one or more stations.

Request Initialization Mode (RIM) Response: This response is used to request that the initialization mode be established.

Request Disconnect (RD) Response: This request is used to request that a link be disconnected.

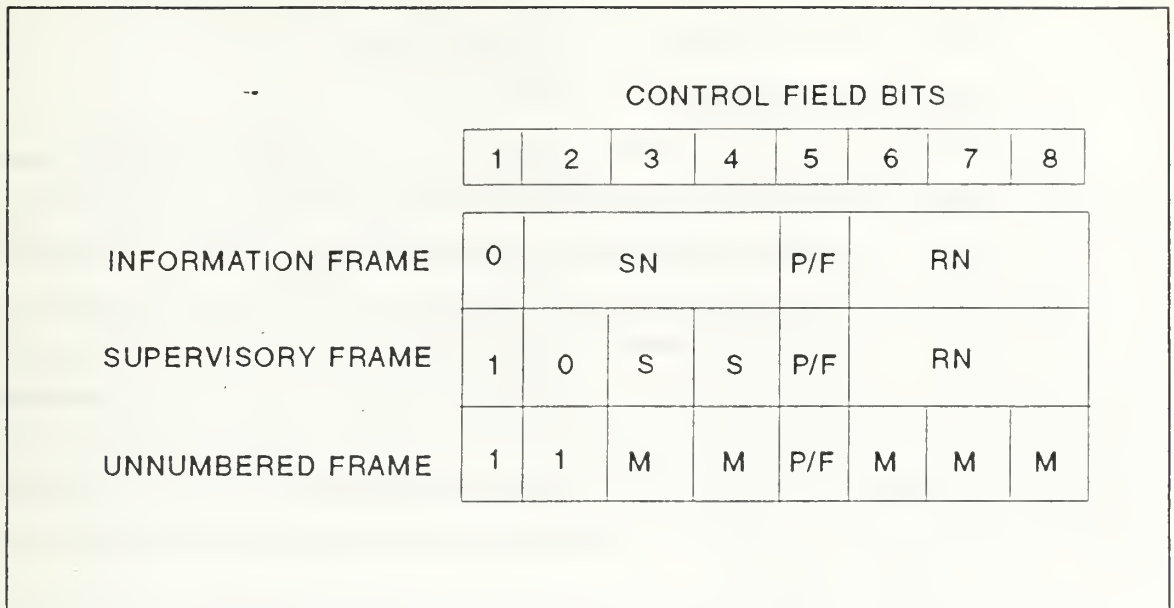


Figure 6. Control field format: [Ref. 8]

Unnumbered Acknowledgment (UA) Response: This response is used to acknowledge receipt and execution of a mode setting, initialization, resetting, or disconnecting command.

Frame Reject (FRMR) Response: This response is used to indicate that a frame received was in error due to a format error.

The information field contains the data packet to be transferred. The length of this field may be variable.

The FCS field is a 16-bit field reserved for an error detection technique. The common parity check codes used are the cyclic redundancy check (CRC) codes. The CRC is used to detect bit errors in the bit sequence between the flag fields.

One method to calculate the CRC bit sequence is as follows. The transmitter performs a long division operation on the frame's bit sequence less the flag fields. The divisor is a generator polynomial. The result of this long division is the CRC bit sequence. There are two types of generator polynomials commonly used [Ref. 7: pp. 54-58]:

CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + 1$

CRC-16 polynomial: $x^{16} + x^{15} + x^2 + 1$

The receiver must perform a similar operation to determine if any errors have occurred. The dividend is the frame's bit sequence less the flag fields. The divisor is the

same generator polynomial. If there is no remainder, the frame was received without errors. If there is a non-zero remainder, the frame is rejected.

C. DATA TRANSFER MODES

The tactical VSAT network will be a multiaccess network with full-duplex transmission. During periods when the communications channel is experiencing high bit error ratios, it may be necessary to change to half-duplex operation. There are three types of data transfer modes of operation that can be implemented to control the network.

The first mode is the normal response mode (NRM). This mode is usually used in networks that are in the star configuration. The hub station controls the entire network. The remote VSAT terminals may transmit data only when the hub station has transmitted polling commands. The NRM is considered reasonable for use with half-duplex links or when the remote stations do not have the capability of interacting with other remote stations. [Ref. 7: p. 85]

The second mode is asynchronous response mode (ARM). This mode is more flexible than the NRM in that data transfer between the hub station and the remote stations can be performed asynchronously without the polling commands. This lowers the requirement of polling overhead and allows data transfer to continue more freely.

The third mode is the asynchronous balanced mode (ABM). With this mode, all stations have equal capability to initiate data transfer with another station and to control the data transfer.

As stated in chapter 2, the tactical VSAT network will be implemented with DAMA and under the control of the master station. The master station will control the scheduling of when VSATs may transmit and how many may transmit at one time. The ABM mode is the most efficient of the three modes and is recommended for this network. Once a channel is reserved for two stations or permission is granted for two stations to communicate to each other, the ABM mode can be initialized and data transfer initiated. A DLC layer protocol must be selected for the tactical VSAT network and implemented to ensure a virtual bit error-free link.

IV. ARQ PROTOCOLS

A. INTRODUCTION

A DLC layer protocol is a specific set of procedures for the transfer of data frames to achieve a virtual error-free link between two stations. As stated above, the DLC protocol must be able to detect errors that occurred during transmission. There is a technique called forward error correction (FEC) which corrects errors without requesting retransmission. However, FEC can not guarantee an error-free link. The standard DLC layer technique is called an automatic repeat request (ARQ) protocol. There are three types of ARQ protocols: Stop-and-Wait, Go-Back-N, and Selective Repeat. In general, they detect frame errors and request the sender to retransmit the frame.

B. STOP-AND-WAIT

The Stop-and-Wait (SW) protocol works on a frame-by-frame basis. A single frame is transmitted and the transmitter has a timeout delay to wait for a response from the receiving station. If the receiving station receives the frame without errors, an acknowledgment (ACK) is returned to the transmitting station requesting for the next frame to be transmitted. If the receiving station detects an errored frame, a request for retransmission or negative acknowledgement (NAK) is returned to the transmitting station. The transmitting station retransmits the same frame until an ACK is received. This technique is used when transmission delays are very short, or when stopping transmission does not cause resynchronization problems. It can not be used in satellite networks because of the long roundtrip propagation delay between a transmitting station and a receiving station.

Therefore, for long transmission delays, the Go-Back-N and Selective Repeat protocols are used. These protocols allow the transmitter to send frames continuously without having to wait for a response for each frame sent. For effectiveness and efficiency, there is a limit on the number of frames which can be sent before the acknowledgment for first transmitted frame is received. This limit is called a window and is determined by the transmission path delay and required system capabilities.

C. GO-BACK-N

Packets from a network layer are passed to the DLC layer in sequential order. The send sequence number SN is sent in the frame header as shown in Figure 6. The Go-

Back-N (GBN) protocol allows a transmitter to transmit packets continuously without waiting for the next packet to be requested. The receiving DLC sends request number RN back to the transmitter requesting packet RN and acknowledging all packets sent before RN. The GBN protocol requires the receiver to receive frames in proper sequence. The receiver looks at only one frame at a time.

The parameter N specifies the number of packets which can be sent successively without a request for a new packet. The transmitter can not transmit packet $(i+N)$ before the packet i is acknowledged (i.e., before the packet $(i+1)$ is requested). Let $w(0)$ represent the number of the last RN that the transmitter received. By the GBN protocol, the transmitter can send packets in the window from $w(0)$ to $w(N-1)$, but not higher numbered packets. The start of the window is denoted by $w(0)$ and the end of the window is denoted by $w(N-1)$. As successively higher packets are requested, $w(0)$ increases and the window slides forward. The window size N and the frame length in bits must be selected such that their product divided by the channel rate is larger than the roundtrip transmission path delay plus processing time at the receiving station. This will ensure that the packet i may be acknowledged before packet $i+N-1$ is transmitted. After packet $i+N-1$ is transmitted, the GBN protocol calls for a timeout delay. The timeout delay is equal to the round trip transmission path delay plus some processing time at the receiving station. If a RN is not received during the timeout delay, the entire window of packets is retransmitted.

As each frame is received at the receiver, the packet's send sequence number SN is checked to see if it is the frame that the receiver has requested. If the SN is correct, the CRC is analyzed to determine if an error has occurred. If the frame was received correctly, a frame (I, S or U frame) containing a request number RN (i.e., $RN = SN + 1$) is sent to the transmitter requesting the next frame. If the frame was received in error, a frame containing a RN equal to the SN is sent to the transmitter to request retransmission of the same frame. The transmitter will retransmit the errored packet and all packets (up to $w(N-1)$) that were transmitted during the time period from the errored packet was first transmitted until the request for retransmission was received. It is important to note here that the receiver will continue to receive frames from the transmitter with increasing SNs. After an errored frame has been detected, the receiver awaits a correct frame with the same SN as the errored frame. All frames, correct or errored, received after the receipt of the errored frame have SNs different from the SN of the errored frame. Consequently, those frames are rejected. The handling of an acknowledgment and a negative acknowledgment are shown in part (a) of Figure 7. Handling of a

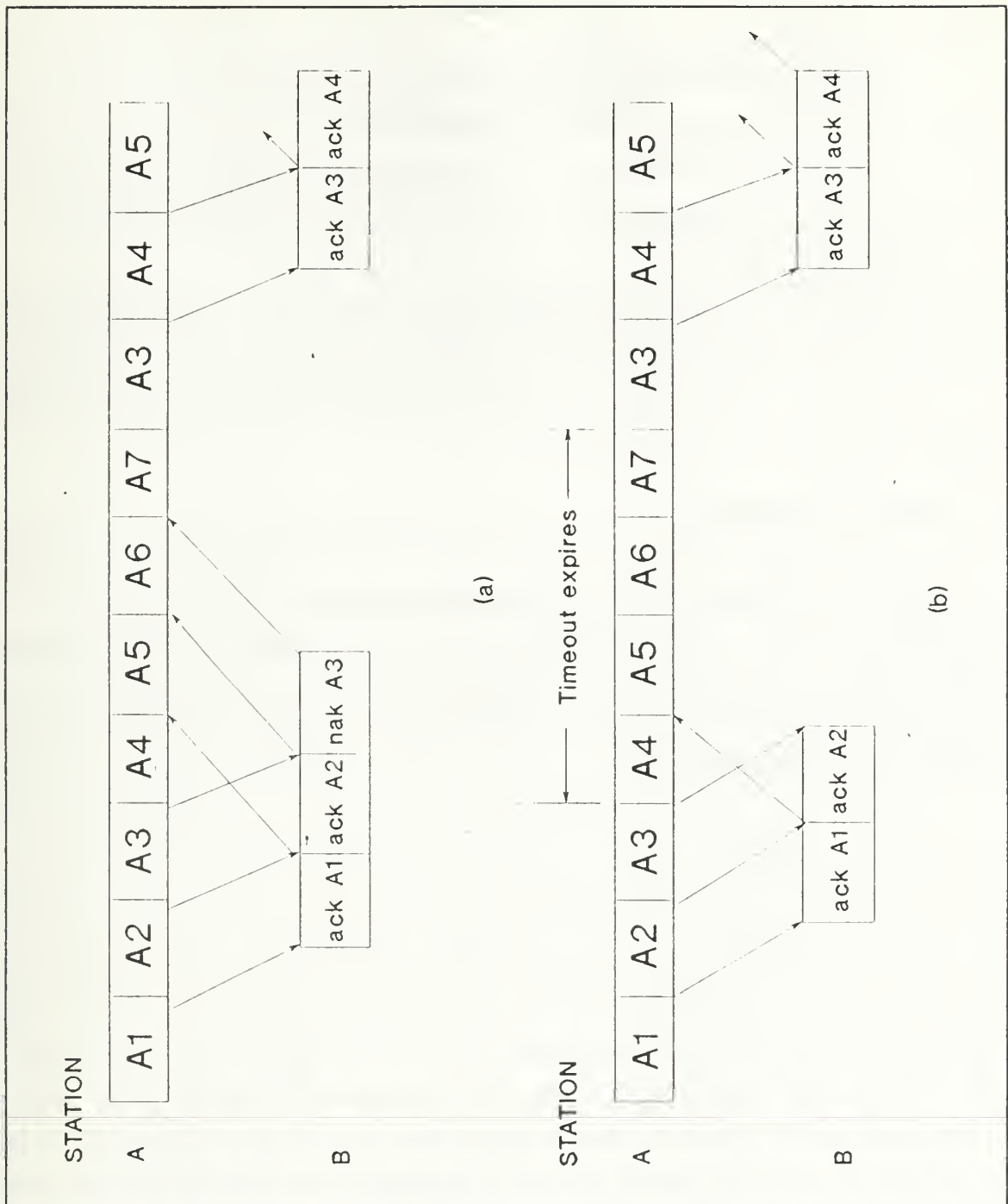


Figure 7. GBN Protocol: (a) handling of ACK/NAK; (b) timeout delay: (Source: Misha Schwartz, *Telecommunication Networks*, p. 123, Addison-Wesley Publishing Co., Reading, Massachusetts, 1987.)

timeout delay is shown in part (b) of Figure 7. For simplicity, data transfer is in one direction only from Station A to Station B.

The GBN protocol is designed for point-to-point full-duplex operation. The two stations involved may send a packet and acknowledge receipt of a packet in the same frame. The I frame format allows both a SN and a RN. Or, if one of the stations does not have any more packets to send, it must still send RNs in S frames to the other station. [Ref. 7: pp. 63-68]

If the communications channel is subjected to a lot of noise, the throughput of the GBN protocol decreases rapidly because numerous frames have to be retransmitted. The noisy communications channel may also affect the RN packets. Remembering that if the transmitting station has not received any RNs after the timeout delay, the entire window of packets is retransmitted.

D. SELECTIVE REPEAT

The Selective Repeat (SR) protocol is similar to the GBN protocol in the use of the sliding window. However, the SR protocol requires only the errored frames to be retransmitted. Therefore, the expected throughput of the SR protocol is higher than the GBN protocol. The complexity of implementing the SR protocol is also much higher in that frames may be received out of order and must be placed in order either in the DLC layer or the network layer.

V. ADAPTIVE GO-BACK-N (GBN) STRATEGY AND IMPLEMENTATION

A. GBN THROUGHPUT EFFICIENCY

The effectiveness of data transfer with the GBN protocol is dependent on the communications channel bit error ratio. The optimum frame size can be determined for any channel with a stable bit error ratio. The optimum frame size is calculated by selecting a threshold for the probability of error for a frame with the length equal to the number of information bits (packet size) plus the number of overhead bits.

$$P_e = 1 - (1 - P_b)^{L+H} \quad (1)$$

where L is the packet length in bits and H is the number of overhead control bits per frame.

With the optimum packet size, the probability of a frame error will be minimized with respect to the threshold. It would seem possible to select a very short frame length to maintain a very low probability of error. However, this action would decrease throughput efficiency drastically. (Throughput efficiency is defined as the number of information data bits transmitted relative to the total number of data bits transmitted.) In the GBN protocol, all frames have the same number of overhead bits regardless of frame size. Therefore, throughput increases as the length of the information field increases. Unfortunately, as the overall frame length is increased, the probability of a frame error increases. Realistically, the maximum throughput efficiency is achieved by calculating the optimum packet length based upon the average channel bit error conditions and by minimizing the number of retransmissions. If the channel bit error ratio changes slowly over long periods of time, it may be feasible to manually change the packet size to maintain maximum throughput efficiency. It would not be feasible to do this if the channel bit error ratio were subject to wide variations. Under these conditions, the GBN protocol would continue to transmit frames that would be received in error. Consequently, throughput efficiency would decrease since the protocol calls for retransmission of all frames transmitted from the time the errored frame was transmitted until the request for retransmission of the frame was received.

The channel bit error ratio for a VSAT satellite network is subject to more variations than a satellite network with larger terminals (i.e., higher transmission power and an-

tennas with larger diameters). The low power VSATs must cope with atmospheric changes (e.g. rain, snow, clouds), interference from adjacent satellite systems, and jamming. Any combination of these types of sources may cause either signal fading, an increase in noise level, or both. The time period that the satellite signal is degraded may be relatively short (bursty) or persist over a long period of time. Therefore, an adaptive GBN strategy which adapts the packet length as the bit error ratio increases may achieve a higher throughput efficiency than the standard GBN protocol.

B. ADAPTIVE GBN STRATEGY

Past improvements of ARQ protocols resulted in a mixture of the three standard ARQ protocols SW, GBN and SR, or some modified version of them. However, many of those protocols retained fixed parameter values for frame length and window size. Some of the improvements varied the packet length or the window size but not both parameters. This prompted two communications engineers in Japan, Mr. Y. Ishibashi and Mr. A. Iwabuchi, to perform a throughput analysis of adaptive ARQ schemes in which both the packet length L and the window size M changed dynamically with the channel bit error ratio. [Ref. 9] The Japanese engineers applied adaptive control to GBN and SR protocols with a half-duplex operations and made an analysis of throughput efficiency. For adaptive GBN with half-duplex operation, the transmitter would transmit $M(0)$ frames, each of length $L(0)$ bits, stop transmission, and wait a specific period of time (timeout delay) for a response from the receiver. The parameters $M(0)$ and $L(0)$ indicate the initial stage (i.e., stage 0) of the adaptive GBN protocol. If the receiver received all the frames correctly, the receiver would send an acknowledgment to the transmitter and the transmitter would send another $M(0)$ frames of frame length $L(0)$. If an errored frame was detected, the receiver would send a negative acknowledgment to the transmitter. The transmitter would transition to stage 1, and retransmit frames starting with the first frame in which an error was detected. The identification of the errored frame is accomplished by assigning sequence numbers to the frames which indicates the order of transmission. Only this time, the frame length would be reduced to $L(1)$ and the window size increased to $M(1)$. The adaptive GBN strategy can be applied with K stages.

In general:

- stage 0: transmit $M(0)$ frames of length $L(0)$;
If all frames are received correctly, remain at stage 0;
Else, transition to stage 1.
- stage 1: transmit $M(i)$ frames of length $L(i)$;

If all frames are received correctly, return to stage $i-1$;
Else, transition to stage $i+1$.

- Continue for

$$1 \leq i \leq K-2$$

- stage $K-1$: transmit $M(K-1)$ frames of length $L(K-1)$;
If all frames are received correctly, return to stage $K-2$;
Else, remain at stage $K-1$.
 $L(i) \geq 1 ; M(i-1) \geq M(i) \geq 1$

Mr. Ishibashi and Mr. Iwabuchi's adaptive GBN strategy was implemented in this thesis with a full-duplex circuit. This modification increases operational efficiency by forwarding immediate acknowledgment or negative acknowledgment to the transmitter. As long as correct frames are received and the receiver sends the acknowledgments, the transmitter continually increments the window index (i.e., sliding the window forward) and does not have to stop transmission and timeout delay until the N th frame is transmitted.

C. ADAPTIVE GBN IMPLEMENTATION

1. General

The key characteristic of the adaptive GBN strategy is the variation of the frame length under varying channel bit error ratios. This is done by increasing or decreasing the length of the packet to be transmitted. Similar to the standard GBN protocol, the transmitting station must store all the transmitted packets until their acknowledgments are received. During periods of increasing bit error ratios, this feature facilitates transitions between the stages by allowing retransmission of the packets at different lengths without having to request the packet from a higher network layer. During periods of decreasing bit error ratios, there is no difficulty in increasing the packet lengths.

The transmitting station must account for several items. The transmitting station must keep track of which packets have been passed, which packets are in the process of transmission, and the beginning packet of the sliding window. Also, the number of acknowledgments for correctly received packets at each stage must be recorded to return to a higher stage (i.e., longer packet length).

Point-to-point communications using the adaptive GBN protocol varies only slightly from communications using the standard GBN protocol. The sessions are initiated in the same manner. The asynchronous balance mode (ABM) permits information flow in both directions. The acknowledgments (ACK) for correctly received packets and negative acknowledgments (NAK) for rejected packets are denoted by the value of RN

in the control field of the frame returned from the receiving station. The packet length and corresponding window size for a stage is selected such that the product of the two divided by the channel bit ratio is slightly greater than the sum of the roundtrip delay and the receiving station's packet processing time. The timeout delay before retransmitting a complete window of packets is equal to the same value.

2. Adaptive GBN Operations and Notation

Typical adaptive GBN operations for a point-to-point communications session are illustrated for a one-way information transfer as shown in Figure 8 and a two-way information transfer, as shown in Figure 9. The examples are for short transmission path delays. For point-to-point communications in a tactical VSAT network, the window size would increase greatly. The asynchronous balanced mode (ABM) has been selected as the data transfer mode. In these examples, the I frame length is specified by the stage. All S and U frames lengths are 48 bits. The occurrence of an error during transmission is denoted by a slash in the slanted lines between stations.

The following frame notation is used to describe protocol operations: [Ref. 7: p.88-91.]

A (func), SN, RN P/F

- **A** represents the frame address. If the frame contains a command, the specified address is the station to receive the frame. If the frame contains a response, the specified address is the station transmitting the response.
- **(func)** represents the abbreviation of the frame's function. For I frames, (func) is 'I'. For S and U frames, examples of the function are represented RR, RNR, SETM, UA, etc.
- **SN** is the send sequence number.
- **RN** is the request sequence number.
- **P/F** represents the P or F flag bit and is shown only when the bit is set.

3. An Example of a One-Way Information Transfer

A one-way point-to-point communications session between Station A and Station B is shown in Figure 8. Information is being transferred from Station A to Station B only. For illustrative purposes only, the transmission path delay is short. The packet lengths and window sizes have been selected such that receipt of the first frame will be acknowledged before the transmitting station completes sending a window's worth of

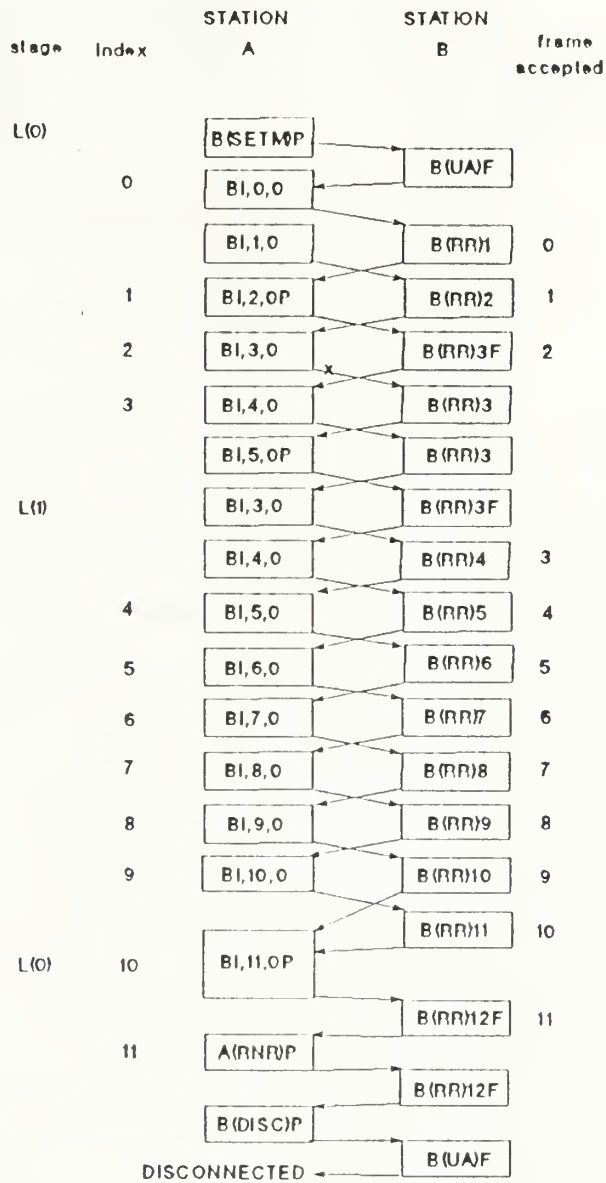


Figure 8. Adaptive GBN: One-way Information Transfer

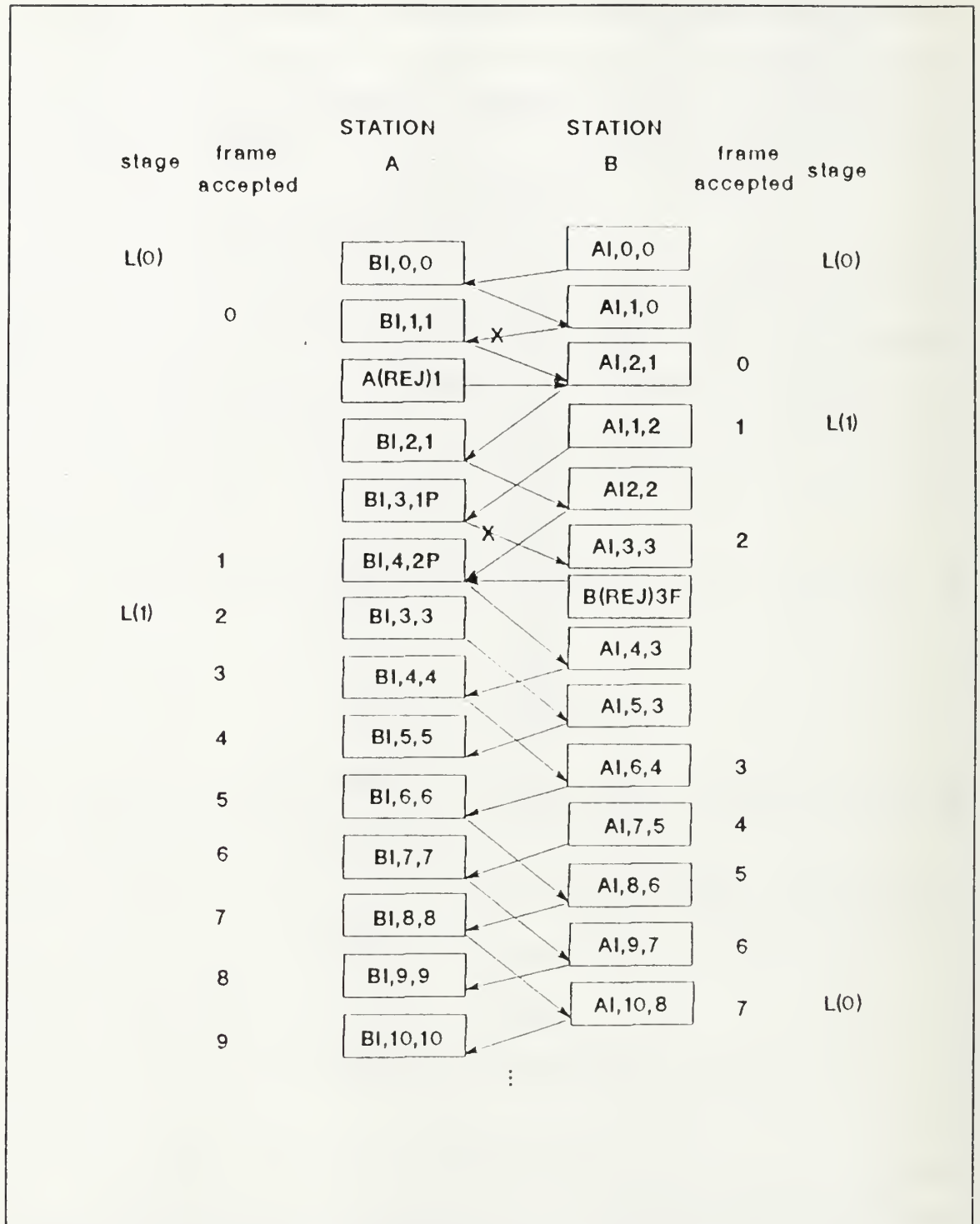


Figure 9. Adaptive GBN: Two-way Information Transfer

packets. Adaptive GBN protocol with two stages will be implemented with the following parameters:

- Stage 0: packet length $L(0) = 1000$ bits; window size $M(0) = 3$
- Stage 1: packet length $L(1) = 500$ bits; window size $M(1) = 6$

Station A initiates the communications session by sending a set mode command (SETM) frame with P bit set. Station B is ready to receive data and responds with an unnumbered acknowledgment (UA) response frame. The UA response frame uses B's address and the F bit is set. Next, B initializes the values of SN and RN to zero. When A receives the UA frame, it also initializes its SN and RN values, and begins transferring data.

Station A transmits frames sequentially according to the adaptive GBN protocol. Station A does not have to wait for each frame to be acknowledged as with the SW protocol. Station B does not have any information to transfer to A. Therefore, acknowledgments for correctly received frames and negative acknowledgments for errored frames are returned to Station A in S or U frames.

Station A transmits 1 frames BI,0,0 and BI,1,0 at frame length $L(0)$. These frames are received correctly and B responds with B(RR)1, and B(RR)2, respectively. The index advances as A transmits a frame and resets as A receives an ACK. The B(RR)2 frame is received by A and processed after the third frame BI,2,0 is transmitted. Station A sends a request for an ACK with the third frame because it has reached the window's limit. Since the B(RR)2 frame is received while the third frame is being transmitted, the index is advanced and Station A can transmit the next frame without waiting for a response from B. An error occurs in the transmission of the fourth frame BI,3,0. Station B sends a NAK to request retransmission of BI,3,0. Meanwhile, Station A continues to transmit BI,4,0 and send a request for ACK with BI,5,0. While BI,5,0 is being transmitted, the NAK for BI,3,0 is processed and BI,3,0 is retransmitted. Station B discards BI,4,0 and BI,5,0 even though they are received without errors because it is expecting BI,3,0. With receipt of the NAK, Station A transitions to stage 1 and retransmits BI,3,0 at a reduced packet length $L(1)$. The example continues with Station A sending packets, and B receiving them correctly. When A has received $M(1)$ number of successive ACKs while in stage 1, it begins the transition back to stage 0. First, A must wait for the acknowledgment of the eleventh frame BI,10,0, the last frame sent with length $L(1)$. Meanwhile, the index has advanced to the 11th frame with the receipt of

B(RR)10. After the ACK is received, A transitions to stage 0 and begin transmitting frame BI,11,0 with packet length $L(0)$.

Finally, A sends the last frame and requests an ACK from B. After B responds, A signals that it does not want to send any more packets by sending a receive not ready (RNR) frame. Station B responds to the poll with an ACK. Station A sends a disconnect command (DISC) to B. Station B acknowledges the disconnect with a UA frame. The session is completed when both stations enter the disconnect mode.

4. An Example of a Two-Way Information Transfer.

A point-to-point communications session is shown in Figure 9. Information is being transferred in both directions between Station A and Station B. For illustrative purposes only, the transmission path delay is short. The packet lengths and window sizes have been selected such that receipt of the first frame will be acknowledged before the transmitting station completes sending a window's worth of packets. Adaptive GBN protocol with two stages will be implemented with the following parameters:

- Stage 0: packet length $L(0) = 1000$ bits; window size $M(0) = 3$
- Stage 1: packet length $L(1) = 500$ bits; window size $M(1) = 6$

For this example, the communication has already been initiated in the ABM and both stations have information to transmit. When both stations have information to transfer, the ACK or NAK for receipt of frames from the opposite station are included in the I frames in the RN field. The adaptive GBN protocol continues to operate as it did with the one-way transfer. Both stations are starting in stage 0.

Station A starts transferring information with BI,0,0 and Station B starts with AI,0,0. Station A receives AI,0,0 as it is transmitting BI,0,0. Station B receives BI,0,0 as it transmits AI,1,0. Station A acknowledges AI,0,0 by setting the RN value to 1, and sends I frame BI,1,1 just as it receives AI,1,0. Frame AI,1,0 was transmitted with improper format. Station A sends a NAK to B using an S frame A(REJ)1. Station B receives BI,1,1 as it is transmitting AI,2,1, the ACK for BI,0,0. Station B responds to the request for retransmission of the second frame, transitions to stage 1, and acknowledges BI,1,1 with an I frame AI,1,2. Station B continues data transfer with AI,2,2. Station A sends BI,2,1 signifying it is transmitting its third I frame and is still expecting B's second I frame. Station A sends BI,2,1 just as it receives AI,2,1, which it discards because it is still expecting B's second frame. Station A receives AI,1,2 just as it sends BI,3,1P, the NAK for AI,2,1 and a poll request because it is at the end of its $M(0)$ window. During transmission of frame BI,3,1P, a large burst of noise causes errors. Station

B receives BI,3,1P as it is sending AI,3,3, the ACK for BI,2,1. The CRC detects the errors in BI,3,1P and rejects the frame. Station B sends a NAK and responds to the poll in the form of an S frame B(REJ)3F. Upon receipt of AI,1,2, Station A increments the window index and sends BI,4,2P, the ACK for AI,1,2, before it receives the REJ. Since A is at the end of the M(0) window again, it also polls B for an ACK. Station B receives BI,4,2P as it is sending AI,4,3, but discards frame BI,4,2P because it is expecting A's fourth frame. Upon receiving the B(REJ)3 frame, Station A transitions to stage 1, responds to the request for retransmission of its fourth frame with BI,3,3, which is also an ACK for AI,2,2.

Now both stations are operating in the adaptive GBN's stage 1. Station A continues data transfer by sending BI,4,4, the ACK for AI,3,3, just as it is receiving AI,4,3. Station B continues to transfer data by sending AI,5,3 just as it receives BI,3,3. Station A send BI,5,5, the ACK for AI,4,3 just as it receives AI,5,3. Station B sends AI,6,4, the ACK for BI,3,3 just as it receives BI,4,4. This process continues until B receives BI,7,7. Station B begins to transition back to stage 0 because it has received an M(1) number of successive ACKs for correct frames received successively by A. Station B must wait for AI,8,6 to be acknowledged before the final transitioning to the stage 0. When Station B receives BI,8,8, it transitions to stage 0 and sends AI,10,8 at packet length L(0).

Now Station A is in stage 1 and Station B is in stage 0. This two-way information transfer will continue until one of the station requests to disconnect and end the session, or until the master station tells both stations to disconnect.

VI. ADAPTIVE GBN SIMULATION

A. INTRODUCTION

The previous work by Mr. Y. Ishibashi and Mr. A. Iwabuchi included an analysis of the throughput efficiency of a type of adaptive GBN protocol as described in this thesis. [Ref. 9] Their throughput analysis of this adaptive protocol was very complex because it is very difficult to statistically model the complete communications session. Since all states and processes cannot be statistically or mathematically modelled completely, various assumptions and preconditions were declared. These assumptions and preconditions were carefully reviewed for any impact on the analysis results.

Another method of analyzing the throughput efficiency of the adaptive GBN protocol is through computer simulation. A computer program must be devised to simulate the adaptive protocol's operations and parameters as closely as possible. A statistical model of the channel must also be possible to investigate the throughput efficiency of the adaptive GBN protocol under increasing bit error conditions. Assumptions and preconditions are also carefully made to minimize their impact on the results. The accuracy of all these items may yield simulation results indicative of actual protocol performance. [Ref. 10.]

B. SIMULATION DEVELOPMENT

1. Model Parameters and Assumptions

The simulation model was designed to simulate the operations of an adaptive GBN protocol during a one-way information transfer communications session between two stations. Since the adaptive GBN is being investigated for use with a tactical VSAT network, the communications channel bit rate was 32,000 bits per second and the transmission path delay is at least 500 milliseconds. The data processing equipment at each station perform the information transfers with negligible delays and are assumed to be dedicated equipments. The satellite path is subject to noise and interference. The bit error ratio for the channel is a precondition statistical parameter and is changed to model different conditions. The packet lengths and window sizes are specified for each stage. The frame overhead is constant. The following conditions are used for building the simulation model:

- Only Station A transfers data. Station B sends ACKs NAKs only.

- Station A's network layer passes packets continuously to the DLC layer.
- The ACKs and NAKs transmitted by Station B are always received without errors because they are very short frames of 48 bits.
- Processing time delays are considered negligible and do not have any impact on throughput efficiency calculations.
- The bit errors which occur during transmission are independent events.

2. Simulation Program

The adaptive GBN protocol simulation was performed with NETWORK 11.5, a CACI Products Company simulation software package. [Ref. 11] NETWORK 11.5 was developed to model computer system configurations and local area networks. Although a VSAT network is considered a wide area network, satellite transmission delays were implemented by specifying a large processing delay at the distant station. NETWORK 11.5 provided enough flexibility to simulate most of the adaptive protocol procedures.

To simulate the two stations, two processing elements (PE) were created. Within each PE, all instructions for protocol operations were defined (e.g., sending frames of different lengths, reading and writing data of different lengths to and from a memory file, receiving frames and acknowledgments, and sending ACKs and NAKs). A storage device (SD) was created to model the network layer with an very long queue of packets to be transferred to the DLC layer. Transfer devices (TD) were created to model the communications channels between the stations. Another TD was created to model the data bus between Station A and the SD. The following parameters were specified for the TDs: bit rate, frame overhead, word size and contention protocol. The bit rate parameter could not be used to model the transmission path delay. Once a frame is identified to be transferred on a TD, the next available module on the receive side is tasked immediately to process the incoming frame. Since the transmission path delay is very long, a different processing module may be available at the frame's actual time of arrival.

The actual simulation of protocol operations is performed by software modules which are dedicated to each PE. Each module performs a list of executable instructions when activated. The procedural flow between modules is similar to a software flow diagram. Preconditions for each module are stated and must be satisfied before the module can execute. The adaptive protocol has its own set of modules associated with data transfer at each stage. While the sets of modules are similar in concept, the instructions are different for handling the different packet lengths and window sizes.

The module preconditions are implemented by declaring message and semaphore preconditions. A message precondition requires that a message must be received before the precondition is satisfied. A semaphore precondition requires that a value of specific parameter must be met before the precondition is satisfied.

Semaphore are parameters which are used as flags to identify the occurrence of an event, and the start or end of a process. Semaphores can be used as counters to be incremented when a semaphore is set and decremented when the semaphore is reset. Proper use of the semaphores aided the troubleshooting of the procedural flow within and between the adaptive protocol's stages.

To simulate the bit error ratio for the TDs, equation 1 was used to determine the probability of frame error P_e based on a given probability of bit error P_b , packet length, and bits of frame overhead. The value of P_e was used by the simulation program to model the transmitter randomly sending frames in error as a percentage of total frames sent. As stated previously, the channel bit error ratio is a precondition. The value of P_b was different for each packet length and for each P_e . A simulation was run for values of P_b in $(10^{-6}, 10^{-2})$.

C. SIMULATION RESULTS

The adaptive GBN protocol was simulated using the simulation model developed above. Three simulation programs were written. One program simulated the standard GBN protocol (constant frame length $L = 1000$ bits and window size $M = 20$). The other two programs simulated the adaptive GBN protocol. One program was written with three stages ($L(0) = 1000$ bits, $M(0) = 20$; $L(1) = 500$ bits, $M(1) = 40$; $L(2) = 250$ bits, $M(2) = 80$). The other program was written with two stages ($L(0) = 1000$ bits, $M(0) = 20$; $L(1) = 250$ bits, $M(1) = 80$) to observe differences in throughput efficiency.

The simulations provided results that were used to calculate the throughput efficiency of each protocol at different bit error ratios. Throughput efficiency is defined as the number of information bits transferred divided by the total number of bits transmitted. The total number of frames transmitted, and the number of frames acknowledged without errors were recorded in the output report files created by NETWORK

11.5.

To calculate the throughput efficiency for the adaptive strategy, the following equations are used:

- Throughput efficiency T (%):

$$T = \frac{\sum_{k=0}^n N_k L(k) E_F(k)}{\sum_{k=0}^n X_k L(k)} \quad (2)$$

where N_k is the number of frames transferred correctly and acknowledged at frame length $L(k)$; X_k is the total number of frames sent at frame length $L(k)$; $E_F(k)$ is the frame efficiency at $L(k)$; the number of stages = $n + 1$.

- Frame efficiency $E_F(k)$ (%):

$$E_F(k) = \frac{L(k)}{L(k) + H} \quad (3)$$

where $H = 48$, the number of overhead bits per frame.

The throughput efficiency results were calculated as described above for the standard GBN protocol and the two adaptive GBN protocols. The results are shown in Figure 11 and Figure 10.

The plot for the adaptive GBN with three stages shows an unexpected result. The adaptive GBN throughput efficiency curve falls below the standard GBN curve between the bit error ratios of 2.0×10^{-5} and 2.5×10^{-4} . Analysis of the data in the bit error ratio range of $(1.0 \times 10^{-6}, 1.0 \times 10^{-5})$ showed the adaptive protocol operated primarily within the first two stages. In this range, the probability of frame error at frame length $L(0)$ increased enough to cause errors occasionally. The protocol transitioned to stage 1 with frame length $L(1)$. The probability of a frame error for a $L(1)$ frame was much lower than the probability of frame error at $L(0)$. Consequently, the protocol was able to reset quickly to stage 0 several times. This resulted in an increased throughput efficiency over the standard GBN protocol.

The bit error ratio was increased to 1.0×10^{-5} . Initially, the probability of frame error at frame length $L(1)$ was low enough to prompt the adaptive protocol to transition to stage 2 occasionally. Consequently, the protocol was able to reset quickly to stage 1. As the bit error ratio increased further, the probability of frame error at frame length

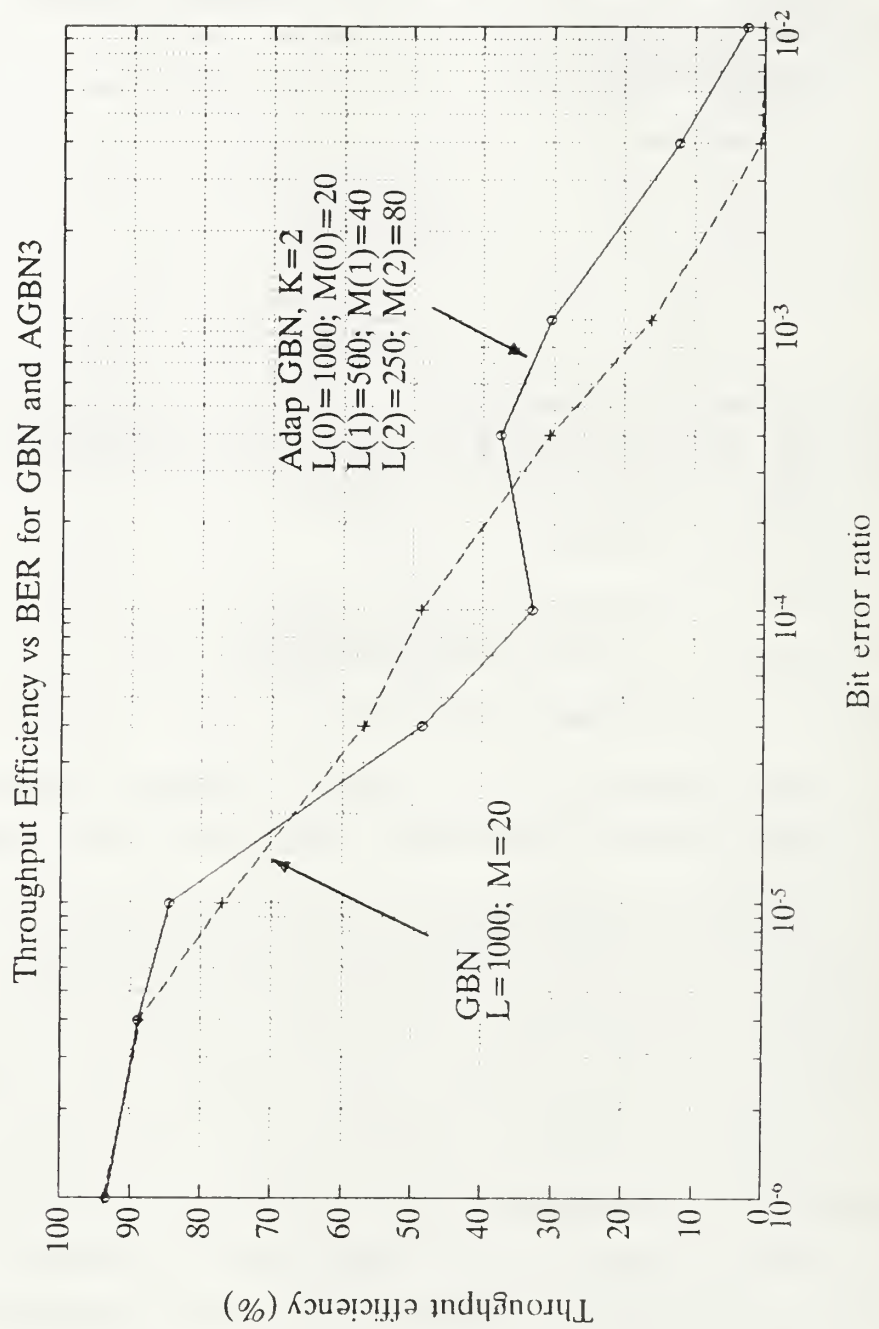


Figure 10. Throughput Efficiency for GBN vs AGBN (3 stages)

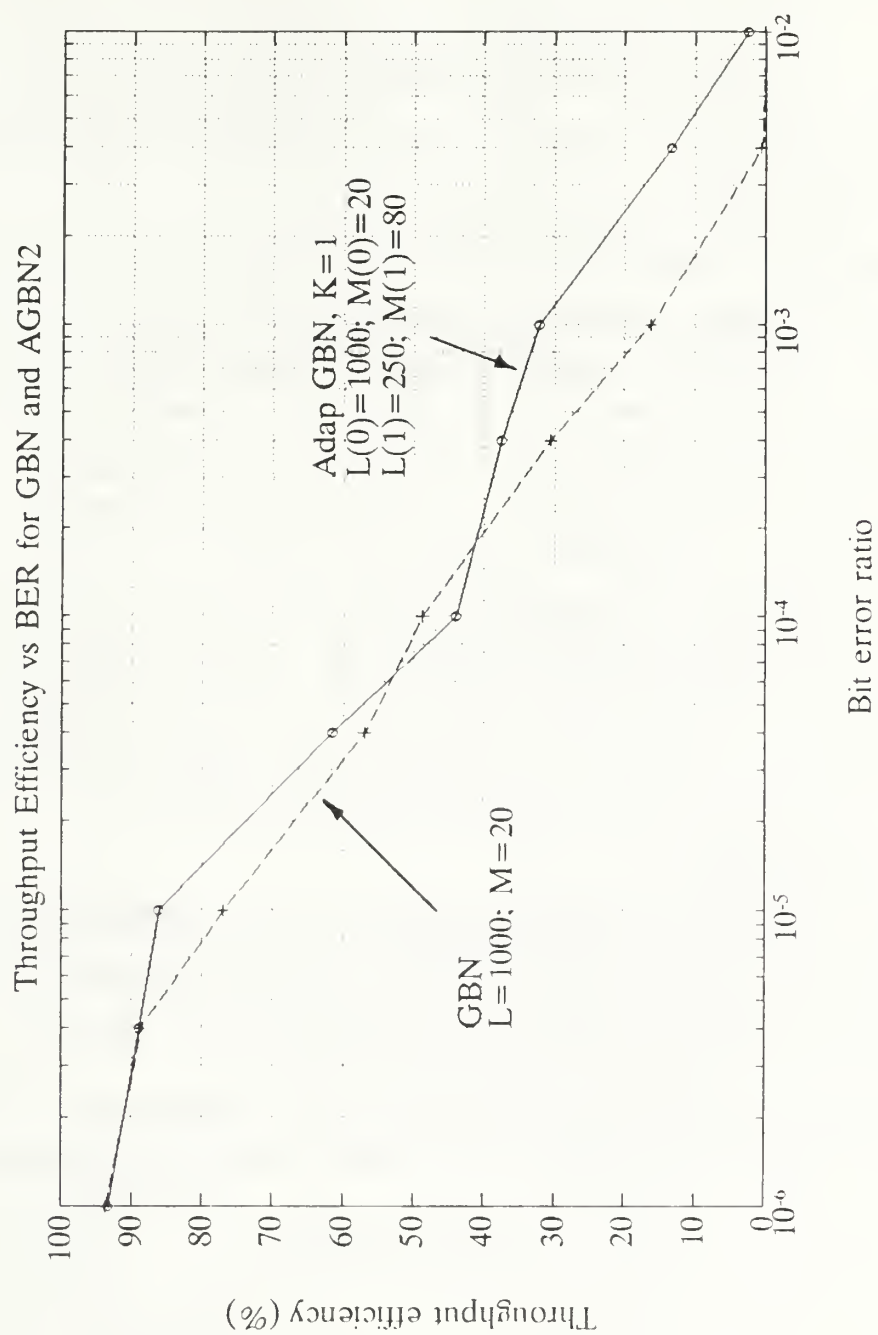


Figure 11. Throughput Efficiency for GBN vs AGBN (2 stages)

$L(1)$ increased enough to cause more frequent transitioning between stage 1 and stage 2. During the transitioning between stages, there were a large number of frames at stage 1 that had to be retransmitted. As an estimate, the number of retransmitted frames would be equal to the window size at stage 1 or $M(1)$ for each transition. The frequent retransmission of $M(1)$ number of frames severely decreased the adaptive protocol's throughput efficiency.

When the bit error ratio increased to 2.5×10^{-4} , the protocol was operating primarily at stage 2 only. This is equivalent to a standard GBN protocol operating with parameters $L(2)$ and $M(2)$. At this higher value of bit error ratio, the probability of frame error at frame length $L(2)$ increased to the point the adaptive protocol could no longer reset back to stage 1. However, the throughput efficiency at the shorter frame length was higher than the throughput efficiency of the standard GBN protocol.

This prompted the simulation of an adaptive GBN protocol with two stages. In short, the three-stage adaptive protocol was trimmed to two stages by eliminating the stage 1 parameters. The throughput efficiency plot for the adaptive GBN protocol with two stages still falls slightly under the standard GBN curve. But, the maximum difference is less than ten percent between the curves over a much smaller range of bit error ratios (6.0×10^{-5} , 2.0×10^{-4}). The adaptive GBN protocol with two stages shows a much better throughput efficiency at bit error ratios greater than 1.0×10^{-4} .

D. CONCLUSION.

The computer simulation of an adaptive GBN protocol with three stages has shown a throughput efficiency problem associated with frequent transitioning between stages. The simulation of the adaptive GBN protocol with two stages has demonstrated a throughput efficiency improvement over a standard GBN protocol over a wide range of bit error ratios. As bit error ratios increased, the throughput efficiency for the adaptive GBN protocol with two stages was much higher than the standard protocol.

APPENDIX A. SIMULATION INPUT FILES

CACI NETWORK 11.5 RELEASE 4.01

This is the input file for an Adaptive GBN protocol with three stages. The probability of bit error has been set to 1.0×10^{-6} . This input file was generated by the CACI NETIN editor and used to run the simulation on CACI NETWORK 11.5. There are two stations, Station A and Station B. There are two 32 kbps channels between them for full duplex operation. The three stages are as follows:

Stage 0: $L(0) = 1000$, $M(0) = 20$

Stage 1: $L(1) = 500$, $M(1) = 40$

Stage 2: $L(2) = 250$, $M(2) = 80$

* agbn3.net

***** GLOBAL FLAGS

GLOBAL FLAGS =

ANTITHETIC VARIATE = NO

RANDOMIZER = 6

NETIN TIME UNITS =

ITERATE BY PRIORITY = NO

CLOCK = YES

BATCH = YES

INPUT LISTING = NO

DEFAULT LISTING = NO

LENGTH = 800.0 SECONDS

PERIODIC REPORTS = 3

INSTRUCTION EXECUTION

SNAPSHOT

PLOT DATA FILE = NO

WIDE REPORTS = NO

TRACE = NO

***** PROCESSING ELEMENTS - SYS. PE. SET

HARDWARE TYPE = PROCESSING

NAME = STATION A

BASIC CYCLE TIME = 1.000000 MICROSEC

INPUT CONTROLLER = YES

MESSAGE LIST SIZE = 640000.0

LOSE OVERFLOW MESSAGES = YES

INSTRUCTION REPERTOIRE =

INSTRUCTION TYPE = READ

NAME ; READ DATA L(0)

STORAGE DEVICE TO ACCESS ; A MEMORY

FILE ACCESSED ; GENERAL STORAGE

NUMBER OF BITS TO TRANSMIT ; 1000

DESTROY FLAG ; YES

RESUME FLAG ; NO

```

ALLOWABLE BUSSES ;
  A BUS
NAME ; READ DATA L(1)
  STORAGE DEVICE TO ACCESS ; A MEMORY
  FILE ACCESSED ; GENERAL STORAGE
  NUMBER OF BITS TO TRANSMIT ; 500
  DESTROY FLAG ; YES
  RESUME FLAG ; YES
  ALLOWABLE BUSSES ;
    A BUS
NAME ; READ DATA L(2)
  STORAGE DEVICE TO ACCESS ; A MEMORY
  FILE ACCESSED ; GENERAL STORAGE
  NUMBER OF BITS TO TRANSMIT ; 250
  DESTROY FLAG ; YES
  RESUME FLAG ; YES
  ALLOWABLE BUSSES ;
    A BUS
INSTRUCTION TYPE = WRITE
NAME ; WRITE WINDOW L(0)
  STORAGE DEVICE TO ACCESS ; A MEMORY
  FILE ACCESSED ; GENERAL STORAGE
  NUMBER OF BITS TO TRANSMIT ; 21000
  REPLACE FLAG ; NO
  RESUME FLAG ; YES
  ALLOWABLE BUSSES ;
    A BUS
NAME ; WRITE WINDOW L(1)
  STORAGE DEVICE TO ACCESS ; A MEMORY
  FILE ACCESSED ; GENERAL STORAGE
  NUMBER OF BITS TO TRANSMIT ; 20500
  REPLACE FLAG ; NO
  RESUME FLAG ; YES
  ALLOWABLE BUSSES ;
    A BUS
NAME ; WRITE DATA L(0)
  STORAGE DEVICE TO ACCESS ; A MEMORY
  FILE ACCESSED ; GENERAL STORAGE
  NUMBER OF BITS TO TRANSMIT ; 1000
  REPLACE FLAG ; NO
  RESUME FLAG ; YES
  ALLOWABLE BUSSES ;
    A BUS
NAME ; WRITE DATA L(1)
  STORAGE DEVICE TO ACCESS ; A MEMORY
  FILE ACCESSED ; GENERAL STORAGE
  NUMBER OF BITS TO TRANSMIT ; 500
  REPLACE FLAG ; NO
  RESUME FLAG ; YES
  ALLOWABLE BUSSES ;
    A BUS
NAME ; WRITE WINDOW L(2)
  STORAGE DEVICE TO ACCESS ; A MEMORY
  FILE ACCESSED ; GENERAL STORAGE
  NUMBER OF BITS TO TRANSMIT ; 20250
  REPLACE FLAG ; NO

```

```

RESUME FLAG ; YES
ALLOWABLE BUSSES ;
  A BUS
NAME ; WRITE DATA L(2)
  STORAGE DEVICE TO ACCESS ; A MEMORY
  FILE ACCESSED ; GENERAL STORAGE
  NUMBER OF BITS TO TRANSMIT ; 250
  REPLACE FLAG ; NO
  RESUME FLAG ; YES
  ALLOWABLE BUSSES ;
    A BUS
INSTRUCTION TYPE = MESSAGE
NAME ; SEND GOOD DATA L(0)
  MESSAGE ; GOOD DATA L(0)
  LENGTH ; 1000 BITS
  DESTINATION PROCESSOR ; STATION B
  QUEUE FLAG ; YES
  RESUME FLAG ; YES
  ALLOWABLE BUSSES ;
    CHANNEL 1
NAME ; SEND BAD DATA L(0)
  MESSAGE ; BAD DATA L(0)
  LENGTH ; 1000 BITS
  DESTINATION PROCESSOR ; STATION B
  QUEUE FLAG ; YES
  RESUME FLAG ; YES
  ALLOWABLE BUSSES ;
    CHANNEL 1
NAME ; SEND GOOD DATA L(1)
  MESSAGE ; GOOD DATA L(1)
  LENGTH ; 500 BITS
  DESTINATION PROCESSOR ; STATION B
  QUEUE FLAG ; YES
  RESUME FLAG ; YES
  ALLOWABLE BUSSES ;
    CHANNEL 1
NAME ; SEND BAD DATA L(1)
  MESSAGE ; BAD DATA L(1)
  LENGTH ; 500 BITS
  DESTINATION PROCESSOR ; STATION B
  QUEUE FLAG ; YES
  RESUME FLAG ; YES
  ALLOWABLE BUSSES ;
    CHANNEL 1
NAME ; SEND GOOD DATA L(2)
  MESSAGE ; GOOD DATA L(2)
  LENGTH ; 250 BITS
  DESTINATION PROCESSOR ; STATION B
  QUEUE FLAG ; YES
  RESUME FLAG ; YES
  ALLOWABLE BUSSES ;
    CHANNEL 1
NAME ; SEND BAD DATA L(2)
  MESSAGE ; BAD DATA L(2)
  LENGTH ; 250 BITS
  DESTINATION PROCESSOR ; STATION B

```

```

        QUEUE FLAG ; YES
        RESUME FLAG ; YES
        ALLOWABLE BUSSES ;
        CHANNEL 1
INSTRUCTION TYPE = PROCESSING
        NAME ; TIMEOUT DELAY
        TIME ; 300000 CYCLES
INSTRUCTION TYPE = SEMAPHORE
        NAME ; SET A MSG XMIT
        SEMAPHORE ; A MSG XMIT
        SET/RESET FLAG ; SET
        NAME ; RESET A MSG XMIT
        SEMAPHORE ; A MSG XMIT
        SET/RESET FLAG ; RESET
        NAME ; SET FN
        SEMAPHORE ; FN
        SET/RESET FLAG ; SET
        NAME ; RESET FN
        SEMAPHORE ; FN
        SET/RESET FLAG ; RESET
        NAME ; SET SN
        SEMAPHORE ; SN
        SET/RESET FLAG ; SET
        NAME ; RESET SN
        SEMAPHORE ; SN
        SET/RESET FLAG ; RESET
        NAME ; SET REXMIT WINDOW
        SEMAPHORE ; REXMIT WINDOW
        SET/RESET FLAG ; SET
        NAME ; RESET REXMIT WINDOW
        SEMAPHORE ; REXMIT WINDOW
        SET/RESET FLAG ; RESET
        NAME ; SET W(0)
        SEMAPHORE ; W(0)
        SET/RESET FLAG ; SET
        NAME ; RESET W(0)
        SEMAPHORE ; W(0)
        SET/RESET FLAG ; RESET
        NAME ; SET ALT RCV
        SEMAPHORE ; ALT RCV
        SET/RESET FLAG ; SET
        NAME ; RESET ALT RCV
        SEMAPHORE ; ALT RCV
        SET/RESET FLAG ; RESET
        NAME ; SET N
        SEMAPHORE ; N
        SET/RESET FLAG ; SET
        NAME ; RESET N
        SEMAPHORE ; N
        SET/RESET FLAG ; RESET
        NAME ; SET PASSCT
        SEMAPHORE ; PASSCT
        SET/RESET FLAG ; SET
        NAME ; RESET PASSCT
        SEMAPHORE ; PASSCT
        SET/RESET FLAG ; RESET

```

```

NAME ; SET RCVCTR FLAG
    SEMAPHORE ; RCVCTR FLAG
    SET/RESET FLAG ; SET
NAME ; RESET RCVCTR FLAG
    SEMAPHORE ; RCVCTR FLAG
    SET/RESET FLAG ; RESET
NAME ; SET RNCOUNT
    SEMAPHORE ; RNCOUNT
    SET/RESET FLAG ; SET
NAME ; RESET RNCOUNT
    SEMAPHORE ; RNCOUNT
    SET/RESET FLAG ; RESET
NAME ; SET RSFLAG
    SEMAPHORE ; RSFLAG
    SET/RESET FLAG ; SET
NAME ; RESET RSFLAG
    SEMAPHORE ; RSFLAG
    SET/RESET FLAG ; RESET
NAME ; SET ALTCTR
    SEMAPHORE ; ALTCTR
    SET/RESET FLAG ; SET
NAME ; RESET ALTCTR
    SEMAPHORE ; ALTCTR
    SET/RESET FLAG ; RESET
NAME ; SET FLAG1
    SEMAPHORE ; FLAG1
    SET/RESET FLAG ; SET
NAME ; RESET FLAG1
    SEMAPHORE ; FLAG1
    SET/RESET FLAG ; RESET
NAME ; SET W(1)
    SEMAPHORE ; W(1)
    SET/RESET FLAG ; SET
NAME ; RESET W(1)
    SEMAPHORE ; W(1)
    SET/RESET FLAG ; RESET
NAME ; START DATA TRANSFER
    SEMAPHORE ; REXMIT WINDOW
    SET/RESET FLAG ; RESET
NAME ; SET L1GOOD
    SEMAPHORE ; L1GOOD
    SET/RESET FLAG ; SET
NAME ; RESET L1GOOD
    SEMAPHORE ; L1GOOD
    SET/RESET FLAG ; RESET
NAME ; SET RN L(0)
    SEMAPHORE ; RN L(0)
    SET/RESET FLAG ; SET
NAME ; SET RN L(1)
    SEMAPHORE ; RN L(1)
    SET/RESET FLAG ; SET
NAME ; SET W(2)
    SEMAPHORE ; W(2)
    SET/RESET FLAG ; SET
NAME ; RESET W(2)
    SEMAPHORE ; W(2)

```

```

    SET/RESET FLAG ; RESET
NAME ; SET L2GOOD
    SEMAPHORE ; L2GOOD
    SET/RESET FLAG ; SET
NAME ; RESET L2GOOD
    SEMAPHORE ; L2GOOD
    SET/RESET FLAG ; RESET
NAME ; SET RN L(2)
    SEMAPHORE ; RN L(2)
    SET/RESET FLAG ; SET
NAME ; RESET RN L(0)
    SEMAPHORE ; RN L(0)
    SET/RESET FLAG ; RESET
NAME ; RESET RN L(1)
    SEMAPHORE ; RN L(1)
    SET/RESET FLAG ; RESET
NAME ; RESET RN L(2)
    SEMAPHORE ; RN L(2)
    SET/RESET FLAG ; RESET
NAME ; SET L(0) RESET FLAG
    SEMAPHORE ; L(0) RESET FLAG
    SET/RESET FLAG ; SET
NAME ; RESET L(0) RESET FLAG
    SEMAPHORE ; L(0) RESET FLAG
    SET/RESET FLAG ; RESET
NAME ; SET L(1) RESET FLAG
    SEMAPHORE ; L(1) RESET FLAG
    SET/RESET FLAG ; SET
NAME ; RESET L(1) RESET FLAG
    SEMAPHORE ; L(1) RESET FLAG
    SET/RESET FLAG ; RESET
NAME = STATION B
    BASIC CYCLE TIME =      1.000000 MICROSEC
    INPUT CONTROLLER = YES
    MESSAGE LIST SIZE = 640000.0
    LOSE OVERFLOW MESSAGES = YES
    INSTRUCTION REPERTOIRE =
        INSTRUCTION TYPE = MESSAGE
            NAME ; SEND RN
                MESSAGE ; RN
                LENGTH ; 6 BITS
                DESTINATION PROCESSOR ; STATION A
                QUEUE FLAG ; YES
                RESUME FLAG ; YES
                ALLOWABLE BUSSES ;
                CHANNEL B
        INSTRUCTION TYPE = PROCESSING
            NAME ; XMSN DELAY
                TIME ; 250000 CYCLES
            NAME ; AWAIT TRAFFIC
                TIME ; 1 CYCLES
        INSTRUCTION TYPE = SEMAPHORE
            NAME ; SET RN XMIT
                SEMAPHORE ; RN XMIT
                SET/RESET FLAG ; SET
            NAME ; RESET RN XMIT

```

SEMAPHORE ; RN XMIT
SET/RESET FLAG ; RESET

***** BUSSES - SYS. BUS. SET

HARDWARE TYPE = DATA TRANSFER

NAME = CHANNEL 1

CYCLE TIME = 31.250000 MICROSEC
BITS PER CYCLE = 1
CYCLES PER WORD = 7
WORDS PER BLOCK = 125
WORD OVERHEAD TIME = 31.250000 MICROSEC
BLOCK OVERHEAD TIME = 1500.000000 MICROSEC
PROTOCOL = FIRST COME FIRST SERVED
BUS CONNECTIONS =
STATION A
STATION B

NAME = A BUS

CYCLE TIME = .050000 MICROSEC
BITS PER CYCLE = 1
CYCLES PER WORD = 7
WORDS PER BLOCK = 125
WORD OVERHEAD TIME = 0. MICROSEC
BLOCK OVERHEAD TIME = .000002 MICROSEC
PROTOCOL = FIRST COME FIRST SERVED
BUS CONNECTIONS =
A MEMORY
STATION A

NAME = CHANNEL B

CYCLE TIME = 31.250000 MICROSEC
BITS PER CYCLE = 1
CYCLES PER WORD = 7
WORDS PER BLOCK = 125
WORD OVERHEAD TIME = 31.250000 MICROSEC
BLOCK OVERHEAD TIME = 1500.000000 MICROSEC
PROTOCOL = FIRST COME FIRST SERVED
BUS CONNECTIONS =
STATION A
STATION B

***** STORAGE DEVICES - SYS. SD. SET

HARDWARE TYPE = STORAGE

NAME = A MEMORY

WORD ACCESS TIME = .1 MICROSEC
BITS PER WORD = 7
WORDS PER BLOCK = 125
READ OVERHEAD TIME PER WORD ACCESS = .1 MICROSEC
WRITE OVERHEAD TIME PER WORD ACCESS = .1 MICROSEC
OVERHEAD TIME PER BLOCK ACCESS = .1 MICROSEC
CAPACITY = 3000000. BITS
NUMBER OF PORTS = 1

***** MODULES - SYS. MODULE. SET

SOFTWARE TYPE = MODULE

NAME = LEVEL 0

PRIORITY = 0
INTERRUPTABILITY FLAG = YES

```

CONCURRENT EXECUTION = NO
ORED PREDECESSOR LIST =
  A START
INSTRUCTION LIST =
  EXECUTE A TOTAL OF ; 1 SET W(0)
ANDED SUCCESSORS =
  CHAIN TO ; PROCESSOR 0
  WITH ITERATIONS THEN CHAIN COUNT OF ;      0
* M(0)=20 L(0)=1000
NAME = PROCESSOR 0
PRIORITY =      0
INTERRUPTABILITY FLAG = YES
CONCURRENT EXECUTION = YES
COMPLETED IF RUN CANCELLED = NO
STATISTICAL SUCCESSOR STREAM =    5
ORED PREDECESSOR LIST =
  LEVEL 0
  XMIT GOOD DATA L(0)
  XMIT BAD DATA L(0)
  REXMIT M(0)
  TIMEOUT DELAY L(0)
  RESET LEVEL 0
REQUIRED SEMAPHORE STATUS =
  CHAIN IF ; FN
  IS ; < 21
  WAIT FOR ; REXMIT WINDOW
  TO BE ; RESET
  WAIT FOR ; ALT RCV
  TO BE ; < 1
  WAIT FOR ; RCVCTR FLAG
  TO BE ; RESET
  WAIT FOR ; RSFLAG
  TO BE ; RESET
  WAIT FOR ; W(0)
  TO BE ; SET
  WAIT FOR ; L(0) RESET FLAG
  TO BE ; RESET
STATISTICAL SUCCESSORS =
  CHOOSE AS SUCCESSOR ; 99.90 % XMIT GOOD DATA L(0)
  CHOOSE AS SUCCESSOR ;  .10 % XMIT BAD DATA L(0)
INSTRUCTION LIST =
  EXECUTE A TOTAL OF ; 1 RESET A MSG XMIT
NAME = XMIT GOOD DATA L(0)
PRIORITY =      0
INTERRUPTABILITY FLAG = YES
CONCURRENT EXECUTION = YES
ANDED PREDECESSOR LIST =
  PROCESSOR 0
REQUIRED SEMAPHORE STATUS =
  WAIT FOR ; A MSG XMIT
  TO BE ; RESET
INSTRUCTION LIST =
  EXECUTE A TOTAL OF ; 1 READ DATA L(0)
  EXECUTE A TOTAL OF ; 1 SEND GOOD DATA L(0)
  EXECUTE A TOTAL OF ; 1 SET SN
  EXECUTE A TOTAL OF ; 1 SET N

```

```

EXECUTE A TOTAL OF ; 1 SET A MSG XMIT
EXECUTE A TOTAL OF ; 1 SET FN
ANDED SUCCESSORS =
CHAIN TO ; PROCESSOR 0
WITH ITERATIONS THEN CHAIN COUNT OF ; 0
CHAIN TO ; TIMEOUT DELAY L(0)
WITH ITERATIONS THEN CHAIN COUNT OF ; 0
NAME = XMIT BAD DATA L(0)
PRIORITY = 0
INTERRUPTABILITY FLAG = YES
CONCURRENT EXECUTION = YES
ANDED PREDECESSOR LIST =
PROCESSOR 0
REQUIRED SEMAPHORE STATUS =
WAIT FOR ; A MSG XMIT
TO BE ; RESET
INSTRUCTION LIST =
EXECUTE A TOTAL OF ; 1 READ DATA L(0)
EXECUTE A TOTAL OF ; 1 SEND BAD DATA L(0)
EXECUTE A TOTAL OF ; 1 SET SN
EXECUTE A TOTAL OF ; 1 SET N
EXECUTE A TOTAL OF ; 1 SET RCVCTR FLAG
EXECUTE A TOTAL OF ; 1 SET A MSG XMIT
EXECUTE A TOTAL OF ; 1 SET FN
ANDED SUCCESSORS =
CHAIN TO ; RCV COUNTER
WITH ITERATIONS THEN CHAIN COUNT OF ; 0
CHAIN TO ; TIMEOUT DELAY L(0)
WITH ITERATIONS THEN CHAIN COUNT OF ; 0
CHAIN TO ; PROCESSOR 0
WITH ITERATIONS THEN CHAIN COUNT OF ; 0
NAME = REXMIT M(0)
PRIORITY = 5
INTERRUPTABILITY FLAG = NO
CONCURRENT EXECUTION = NO
ANDED PREDECESSOR LIST =
REXMIT WINDOW M(0)
REQUIRED SEMAPHORE STATUS =
WAIT FOR ; REXMIT WINDOW
TO BE ; SET
INSTRUCTION LIST =
EXECUTE A TOTAL OF ; 1 WRITE WINDOW L(0)
EXECUTE A TOTAL OF ; 21 RESET FN
EXECUTE A TOTAL OF ; 21 RESET SN
EXECUTE A TOTAL OF ; 21 RESET N
EXECUTE A TOTAL OF ; 21 RESET PASSCT
EXECUTE A TOTAL OF ; 1 RESET REXMIT WINDOW
ANDED SUCCESSORS =
CHAIN TO ; PROCESSOR 0
WITH ITERATIONS THEN CHAIN COUNT OF ; 0
NAME = B START
PRIORITY = 9
INTERRUPTABILITY FLAG = NO
CONCURRENT EXECUTION = NO
START TIME = 0.0
ALLOWED PROCESSORS =

```

```

STATION B
INSTRUCTION LIST =
  EXECUTE A TOTAL OF ; 1 AWAIT TRAFFIC
ANDED SUCCESSORS =
  CHAIN TO ; RN PROCESSOR
  WITH ITERATIONS THEN CHAIN COUNT OF ;      0
NAME = XMIT RN
  PRIORITY =      0
  INTERRUPTABILITY FLAG = NO
  CONCURRENT EXECUTION = NO
  ORED PREDECESSOR LIST =
    RN PROCESSOR
  REQUIRED SEMAPHORE STATUS =
    WAIT FOR ; RN XMIT
    TO BE ; SET
INSTRUCTION LIST =
  EXECUTE A TOTAL OF ; 1 XMSN DELAY
  EXECUTE A TOTAL OF ; 1 SEND RN
  EXECUTE A TOTAL OF ; 1 RESET RN XMIT
ANDED SUCCESSORS =
  CHAIN TO ; RN PROCESSOR
  WITH ITERATIONS THEN CHAIN COUNT OF ;      0
NAME = REXMIT WINDOW M(0)
  PRIORITY =      5
  INTERRUPTABILITY FLAG = NO
  CONCURRENT EXECUTION = NO
  ORED PREDECESSOR LIST =
    TIMEOUT DELAY L(0)
  REQUIRED SEMAPHORE STATUS =
    CHAIN IF ; FN
    IS ; > 20
    WAIT FOR ; W(0)
    TO BE ; SET
INSTRUCTION LIST =
  EXECUTE A TOTAL OF ; 1 SET REXMIT WINDOW
  EXECUTE A TOTAL OF ; 21 SET ALTCTR
  EXECUTE A TOTAL OF ; 1 SET ALT RCV
ANDED SUCCESSORS =
  CHAIN TO ; REXMIT M(0)
  WITH ITERATIONS THEN CHAIN COUNT OF ;      0
NAME = FLOW CONTROL RN
  PRIORITY =      1
  INTERRUPTABILITY FLAG = NO
  CONCURRENT EXECUTION = NO
  START TIME = 0.0
  ALLOWED PROCESSORS =
    STATION A
  REQUIRED MESSAGES =
    RN
  REQUIRED SEMAPHORE STATUS =
    RUN WHEN ; ALT RCV
    IS ; < 1
INSTRUCTION LIST =
  EXECUTE A TOTAL OF ; 1 SET RNCOUNT
  EXECUTE A TOTAL OF ; 1 RESET PASSCT
  EXECUTE A TOTAL OF ; 1 RESET FN

```

```

EXECUTE A TOTAL OF ; 1 RESET N
ANDED SUCCESSORS =
CHAIN TO ; SET ALT RCV
WITH ITERATIONS THEN CHAIN COUNT OF ;      0
CHAIN TO ; RESET LEVEL L(0)
WITH ITERATIONS THEN CHAIN COUNT OF ;      0
CHAIN TO ; RN L(0) COUNTER
WITH ITERATIONS THEN CHAIN COUNT OF ;      0
CHAIN TO ; RN L(1) COUNTER
WITH ITERATIONS THEN CHAIN COUNT OF ;      0
CHAIN TO ; RN L(2) COUNTER
WITH ITERATIONS THEN CHAIN COUNT OF ;      0
CHAIN TO ; RESET LEVEL L(1)
WITH ITERATIONS THEN CHAIN COUNT OF ;      0
NAME = A START
PRIORITY =      9
INTERRUPTABILITY FLAG = NO
CONCURRENT EXECUTION = NO
START TIME = 0.0
ALLOWED PROCESSORS =
STATION A
INSTRUCTION LIST =
EXECUTE A TOTAL OF ; 1 START DATA TRANSFER
EXECUTE A TOTAL OF ; 1 RESET ALT RCV
EXECUTE A TOTAL OF ; 1 RESET RSFLAG
EXECUTE A TOTAL OF ; 1 RESET FLAG1
EXECUTE A TOTAL OF ; 1 RESET L(0) RESET FLAG
EXECUTE A TOTAL OF ; 1 RESET L(1) RESET FLAG
ANDED SUCCESSORS =
CHAIN TO ; LEVEL 0
WITH ITERATIONS THEN CHAIN COUNT OF ;      0
NAME = ALT FLOW CONTROL
PRIORITY =      5
INTERRUPTABILITY FLAG = YES
CONCURRENT EXECUTION = NO
START TIME = 0.0
ALLOWED PROCESSORS =
STATION A
REQUIRED MESSAGES =
RN
REQUIRED SEMAPHORE STATUS =
WAIT FOR ; ALT RCV
TO BE ; > 0
RUN WHEN ; ALT RCV
IS ; > 0
INSTRUCTION LIST =
EXECUTE A TOTAL OF ; 1 RESET ALTCTR
ANDED SUCCESSORS =
CHAIN TO ; ALT 0
WITH ITERATIONS THEN CHAIN COUNT OF ;      0
NAME = ALT 0
PRIORITY =      6
INTERRUPTABILITY FLAG = NO
CONCURRENT EXECUTION = NO
ORED PREDECESSOR LIST =
ALT FLOW CONTROL

```

```

REQUIRED SEMAPHORE STATUS =
  CHAIN IF ; ALTCTR
  IS ; < 1
  WAIT FOR ; ALTCTR
  TO BE ; < 1
INSTRUCTION LIST =
  EXECUTE A TOTAL OF ; 1 RESET ALT RCV
  EXECUTE A TOTAL OF ; 1 RESET RCVCTR FLAG
  EXECUTE A TOTAL OF ; 1 RESET N
  EXECUTE A TOTAL OF ; 1 RESET FLAG1
NAME = RESTORE L(0)
PRIORITY = 3
INTERRUPTABILITY FLAG = NO
CONCURRENT EXECUTION = NO
COMPLETED IF RUN CANCELLED = YES
ORED PREDECESSOR LIST =
  RESTORE L(0)
  SET ALT RCV
REQUIRED SEMAPHORE STATUS =
  CHAIN IF ; FN
  IS ; > 0
  CHAIN IF ; W(0)
  IS ; SET
INSTRUCTION LIST =
  EXECUTE A TOTAL OF ; 1 WRITE DATA L(0)
  EXECUTE A TOTAL OF ; 1 RESET FN
  EXECUTE A TOTAL OF ; 1 RESET SN
  EXECUTE A TOTAL OF ; 1 SET ALTCTR
  EXECUTE A TOTAL OF ; 1 RESET N
ANDED SUCCESSORS =
  CHAIN TO ; RSDONE L(0)
  WITH ITERATIONS THEN CHAIN COUNT OF ; 0
  CHAIN TO ; RESTORE L(0)
  WITH ITERATIONS THEN CHAIN COUNT OF ; 0
NAME = RSDONE L(0)
PRIORITY = 3
INTERRUPTABILITY FLAG = NO
CONCURRENT EXECUTION = NO
ORED PREDECESSOR LIST =
  RESTORE L(0)
REQUIRED SEMAPHORE STATUS =
  CHAIN IF ; FN
  IS ; 0
INSTRUCTION LIST =
  EXECUTE A TOTAL OF ; 1 RESET N
  EXECUTE A TOTAL OF ; 1 RESET RSFLAG
  EXECUTE A TOTAL OF ; 1 RESET FLAG1
  EXECUTE A TOTAL OF ; 1 RESET RN L(0)
ANDED SUCCESSORS =
  CHAIN TO ; LEVEL 1
  WITH ITERATIONS THEN CHAIN COUNT OF ; 0
NAME = SET RCV COUNTER
PRIORITY = 1
INTERRUPTABILITY FLAG = NO
CONCURRENT EXECUTION = NO
ORED PREDECESSOR LIST =

```

```

    SET RCV COUNTER
    RCV COUNTER
REQUIRED SEMAPHORE STATUS =
    CHAIN IF ; N
    IS ; > 0
INSTRUCTION LIST =
    EXECUTE A TOTAL OF ; 1 SET PASSCT
    EXECUTE A TOTAL OF ; 1 RESET N
ANDED SUCCESSORS =
    CHAIN TO ; SET RCV COUNTER
    WITH ITERATIONS THEN CHAIN COUNT OF ;      0
    CHAIN TO ; RESUME XMIT
    WITH ITERATIONS THEN CHAIN COUNT OF ;      0
NAME = RCV COUNTER
    PRIORITY =      1
    INTERRUPTABILITY FLAG = NO
    CONCURRENT EXECUTION = NO
    ORED PREDECESSOR LIST =
        XMIT BAD DATA L(0)
        XMIT BAD DATA L(1)
        XMIT BAD DATA L(2)
REQUIRED SEMAPHORE STATUS =
    WAIT FOR ; RCVCTR FLAG
    TO BE ; SET
    CHAIN IF ; FLAG1
    IS ; RESET
INSTRUCTION LIST =
    EXECUTE A TOTAL OF ; 1 SET FLAG1
ANDED SUCCESSORS =
    CHAIN TO ; SET RCV COUNTER
    WITH ITERATIONS THEN CHAIN COUNT OF ;      0
NAME = RESUME XMIT
    PRIORITY =      1
    INTERRUPTABILITY FLAG = NO
    CONCURRENT EXECUTION = NO
    ORED PREDECESSOR LIST =
        SET RCV COUNTER
REQUIRED SEMAPHORE STATUS =
    CHAIN IF ; N
    IS ; < 1
INSTRUCTION LIST =
    EXECUTE A TOTAL OF ; 1 RESET RCVCTR FLAG
NAME = SET ALT RCV
    PRIORITY =      1
    INTERRUPTABILITY FLAG = NO
    CONCURRENT EXECUTION = NO
    ORED PREDECESSOR LIST =
        FLOW CONTROL RN
REQUIRED SEMAPHORE STATUS =
    CHAIN IF ; PASSCT
    IS ; < 1
    CHAIN IF ; FLAG1
    IS ; SET
INSTRUCTION LIST =
    EXECUTE A TOTAL OF ; 1 SET ALT RCV
    EXECUTE A TOTAL OF ; 1 SET RSFLAG

```

```

EXECUTE A TOTAL OF ; 40 RESET L1GOOD
EXECUTE A TOTAL OF ; 80 RESET L2GOOD
EXECUTE A TOTAL OF ; 1 RESET RNCOUNT
ANDED SUCCESSORS =
CHAIN TO ; RESTORE L(0)
WITH ITERATIONS THEN CHAIN COUNT OF ; 0
CHAIN TO ; RESET ALT RCV L(0)
WITH ITERATIONS THEN CHAIN COUNT OF ; 0
CHAIN TO ; RESTORE L(1)
WITH ITERATIONS THEN CHAIN COUNT OF ; 0
CHAIN TO ; RESTORE L(2)
WITH ITERATIONS THEN CHAIN COUNT OF ; 0
CHAIN TO ; RESET ALT RCV L(1)
WITH ITERATIONS THEN CHAIN COUNT OF ; 0
CHAIN TO ; RESET ALT RCV L(2)
WITH ITERATIONS THEN CHAIN COUNT OF ; 0
NAME = RN PROCESSOR
PRIORITY = 0
INTERRUPTABILITY FLAG = NO
CONCURRENT EXECUTION = NO
ORED PREDECESSOR LIST =
B START
XMIT RN
REQUIRED MESSAGES =
DATA*
REQUIRED SEMAPHORE STATUS =
WAIT FOR ; RN XMIT
TO BE ; RESET
INSTRUCTION LIST =
EXECUTE A TOTAL OF ; 1 SET RN XMIT
ANDED SUCCESSORS =
CHAIN TO ; XMIT RN
WITH ITERATIONS THEN CHAIN COUNT OF ; 0
NAME = RESET ALT RCV L(0)
PRIORITY = 3
INTERRUPTABILITY FLAG = NO
CONCURRENT EXECUTION = NO
ORED PREDECESSOR LIST =
SET ALT RCV
REQUIRED SEMAPHORE STATUS =
CHAIN IF ; FN
IS ; < 1
CHAIN IF ; W(0)
IS ; SET
INSTRUCTION LIST =
EXECUTE A TOTAL OF ; 1 RESET ALT RCV
EXECUTE A TOTAL OF ; 1 RESET FLAG1
EXECUTE A TOTAL OF ; 1 RESET RSFLAG
EXECUTE A TOTAL OF ; 1 RESET RCVCTR FLAG
EXECUTE A TOTAL OF ; 1 RESET N
EXECUTE A TOTAL OF ; 1 RESET RN L(0)
NAME = LEVEL 1
PRIORITY = 0
INTERRUPTABILITY FLAG = YES
CONCURRENT EXECUTION = NO
ORED PREDECESSOR LIST =

```

```

RSDONE L(0)
INSTRUCTION LIST =
  EXECUTE A TOTAL OF ; 1 RESET W(0)
  EXECUTE A TOTAL OF ; 1 SET W(1)
ANDED SUCCESSORS =
  CHAIN TO ; PROCESSOR 1
  WITH ITERATIONS THEN CHAIN COUNT OF ;      0
NAME = PROCESSOR 1
  PRIORITY =      0
  INTERRUPTABILITY FLAG = YES
  CONCURRENT EXECUTION = YES
  COMPLETED IF RUN CANCELLED = NO
  STATISTICAL SUCCESSOR STREAM =      5
  ORED PREDECESSOR LIST =
    LEVEL 1
    XMIT GOOD DATA L(1)
    XMIT BAD DATA L(1)
    REXMIT M(1)
    TIMEOUT DELAY L(1)
    RESET LEVEL 1
REQUIRED SEMAPHORE STATUS =
  CHAIN IF ; FN
  IS ; < 41
  WAIT FOR ; REXMIT WINDOW
  TO BE ; RESET
  WAIT FOR ; ALT RCV
  TO BE ; < 1
  WAIT FOR ; RCVCTR FLAG
  TO BE ; RESET
  WAIT FOR ; RSFLAG
  TO BE ; RESET
  WAIT FOR ; W(1)
  TO BE ; SET
  CHAIN IF ; L1GOOD
  IS ; < 40
  WAIT FOR ; L(1) RESET FLAG
  TO BE ; RESET
STATISTICAL SUCCESSORS =
  CHOOSE AS SUCCESSOR ; 99.95 % XMIT GOOD DATA L(1)
  CHOOSE AS SUCCESSOR ; .05 % XMIT BAD DATA L(1)
INSTRUCTION LIST =
  EXECUTE A TOTAL OF ; 1 RESET A MSG XMIT
NAME = XMIT GOOD DATA L(1)
  PRIORITY =      0
  INTERRUPTABILITY FLAG = YES
  CONCURRENT EXECUTION = YES
  ANDED PREDECESSOR LIST =
    PROCESSOR 1
REQUIRED SEMAPHORE STATUS =
  WAIT FOR ; A MSG XMIT
  TO BE ; RESET
INSTRUCTION LIST =
  EXECUTE A TOTAL OF ; 1 READ DATA L(1)
  EXECUTE A TOTAL OF ; 1 SEND GOOD DATA L(1)
  EXECUTE A TOTAL OF ; 1 SET SN
  EXECUTE A TOTAL OF ; 1 SET N

```

```

EXECUTE A TOTAL OF ; 1 SET A MSG XMIT
EXECUTE A TOTAL OF ; 1 SET L1GOOD
EXECUTE A TOTAL OF ; 1 SET FN
ANDED SUCCESSORS =
CHAIN TO ; PROCESSOR 1
WITH ITERATIONS THEN CHAIN COUNT OF ; 0
CHAIN TO ; TIMEOUT DELAY L(1)
WITH ITERATIONS THEN CHAIN COUNT OF ; 0
CHAIN TO ; RESET LEVEL 0
WITH ITERATIONS THEN CHAIN COUNT OF ; 0
NAME = XMIT BAD DATA L(1)
PRIORITY = 0
INTERRUPTABILITY FLAG = YES
CONCURRENT EXECUTION = YES
ANDED PREDECESSOR LIST =
PROCESSOR 1
REQUIRED SEMAPHORE STATUS =
WAIT FOR ; A MSG XMIT
TO BE ; RESET
INSTRUCTION LIST =
EXECUTE A TOTAL OF ; 1 READ DATA L(1)
EXECUTE A TOTAL OF ; 1 SEND BAD DATA L(1)
EXECUTE A TOTAL OF ; 1 SET SN
EXECUTE A TOTAL OF ; 1 SET N
EXECUTE A TOTAL OF ; 1 SET RCVCTR FLAG
EXECUTE A TOTAL OF ; 1 SET A MSG XMIT
EXECUTE A TOTAL OF ; 1 SET FN
EXECUTE A TOTAL OF ; 40 RESET L1GOOD
ANDED SUCCESSORS =
CHAIN TO ; RCV COUNTER
WITH ITERATIONS THEN CHAIN COUNT OF ; 0
CHAIN TO ; TIMEOUT DELAY L(1)
WITH ITERATIONS THEN CHAIN COUNT OF ; 0
CHAIN TO ; PROCESSOR 1
WITH ITERATIONS THEN CHAIN COUNT OF ; 0
NAME = REXMIT WINDOW M(1)
PRIORITY = 5
INTERRUPTABILITY FLAG = NO
CONCURRENT EXECUTION = NO
ORED PREDECESSOR LIST =
TIMEOUT DELAY L(1)
REQUIRED SEMAPHORE STATUS =
CHAIN IF ; FN
IS ; > 40
WAIT FOR ; W(1)
TO BE ; SET
INSTRUCTION LIST =
EXECUTE A TOTAL OF ; 1 SET REXMIT WINDOW
EXECUTE A TOTAL OF ; 41 SET ALTCTR
EXECUTE A TOTAL OF ; 1 SET ALT RCV
ANDED SUCCESSORS =
CHAIN TO ; REXMIT M(1)
WITH ITERATIONS THEN CHAIN COUNT OF ; 0
NAME = REXMIT M(1)
PRIORITY = 5
INTERRUPTABILITY FLAG = NO

```

```

CONCURRENT EXECUTION = NO
ANDED PREDECESSOR LIST =
    REXMIT WINDOW M(1)
REQUIRED SEMAPHORE STATUS =
    WAIT FOR ; REXMIT WINDOW
    TO BE ; SET
INSTRUCTION LIST =
    EXECUTE A TOTAL OF ; 1 WRITE WINDOW L(1)
    EXECUTE A TOTAL OF ; 41 RESET FN
    EXECUTE A TOTAL OF ; 41 RESET SN
    EXECUTE A TOTAL OF ; 41 RESET N
    EXECUTE A TOTAL OF ; 41 RESET PASSCT
    EXECUTE A TOTAL OF ; 41 RESET LIGOOD
    EXECUTE A TOTAL OF ; 1 RESET REXMIT WINDOW
ANDED SUCCESSORS =
    CHAIN TO ; PROCESSOR 1
    WITH ITERATIONS THEN CHAIN COUNT OF ;      0
NAME = RESTORE L(1)
PRIORITY =      3
INTERRUPTABILITY FLAG = NO
CONCURRENT EXECUTION = NO
COMPLETED IF RUN CANCELLED = YES
ORED PREDECESSOR LIST =
    SET ALT RCV
    RESTORE L(1)
REQUIRED SEMAPHORE STATUS =
    CHAIN IF ; FN
    IS ; > 0
    CHAIN IF ; W(1)
    IS ; SET
INSTRUCTION LIST =
    EXECUTE A TOTAL OF ; 1 WRITE DATA L(1)
    EXECUTE A TOTAL OF ; 1 RESET FN
    EXECUTE A TOTAL OF ; 1 RESET SN
    EXECUTE A TOTAL OF ; 1 SET ALTCTR
    EXECUTE A TOTAL OF ; 1 RESET N
ANDED SUCCESSORS =
    CHAIN TO ; RSDONE L(1)
    WITH ITERATIONS THEN CHAIN COUNT OF ;      0
    CHAIN TO ; RESTORE L(1)
    WITH ITERATIONS THEN CHAIN COUNT OF ;      0
NAME = RSDONE L(1)
PRIORITY =      3
INTERRUPTABILITY FLAG = NO
CONCURRENT EXECUTION = NO
ORED PREDECESSOR LIST =
    RESTORE L(1)
REQUIRED SEMAPHORE STATUS =
    CHAIN IF ; FN
    IS ; 0
INSTRUCTION LIST =
    EXECUTE A TOTAL OF ; 1 RESET N
    EXECUTE A TOTAL OF ; 1 RESET RSFLAG
    EXECUTE A TOTAL OF ; 1 RESET FLAG1
    EXECUTE A TOTAL OF ; 1 RESET RN L(1)
ANDED SUCCESSORS =

```

```

CHAIN TO ; LEVEL 2
WITH ITERATIONS THEN CHAIN COUNT OF ;      0
NAME = TIMEOUT DELAY L(0)
PRIORITY =      0
INTERRUPTABILITY FLAG = YES
CONCURRENT EXECUTION = NO
ORED PREDECESSOR LIST =
    XMIT GOOD DATA L(0)
    XMIT BAD DATA L(0)
REQUIRED SEMAPHORE STATUS =
    CHAIN IF ; FN
    IS ; > 20
INSTRUCTION LIST =
    EXECUTE A TOTAL OF ; 1 TIMEOUT DELAY
ANDED SUCCESSORS =
    CHAIN TO ; REXMIT WINDOW M(0)
    WITH ITERATIONS THEN CHAIN COUNT OF ;      0
    CHAIN TO ; PROCESSOR 0
    WITH ITERATIONS THEN CHAIN COUNT OF ;      0
NAME = TIMEOUT DELAY L(1)
PRIORITY =      0
INTERRUPTABILITY FLAG = YES
CONCURRENT EXECUTION = NO
ORED PREDECESSOR LIST =
    XMIT GOOD DATA L(1)
    XMIT BAD DATA L(1)
REQUIRED SEMAPHORE STATUS =
    CHAIN IF ; FN
    IS ; > 40
INSTRUCTION LIST =
    EXECUTE A TOTAL OF ; 1 TIMEOUT DELAY
ANDED SUCCESSORS =
    CHAIN TO ; REXMIT WINDOW M(1)
    WITH ITERATIONS THEN CHAIN COUNT OF ;      0
    CHAIN TO ; PROCESSOR 1
    WITH ITERATIONS THEN CHAIN COUNT OF ;      0
NAME = RESET LEVEL 0
PRIORITY =      5
INTERRUPTABILITY FLAG = NO
CONCURRENT EXECUTION = NO
ORED PREDECESSOR LIST =
    XMIT GOOD DATA L(1)
REQUIRED SEMAPHORE STATUS =
    CHAIN IF ; L1GOOD
    IS ; > 39
    CHAIN IF ; FN
    IS ; < 41
INSTRUCTION LIST =
    EXECUTE A TOTAL OF ; 1 SET L(0) RESET FLAG
    EXECUTE A TOTAL OF ; 40 RESET L1GOOD
ANDED SUCCESSORS =
    CHAIN TO ; PROCESSOR 0
    WITH ITERATIONS THEN CHAIN COUNT OF ;      0
NAME = RESET LEVEL L(0)
PRIORITY =      1
INTERRUPTABILITY FLAG = NO

```

```

CONCURRENT EXECUTION = NO
ORED PREDECESSOR LIST =
    FLOW CONTROL RN
REQUIRED SEMAPHORE STATUS =
    CHAIN IF ; L(0) RESET FLAG
    IS ; SET
    CHAIN IF ; FN
    IS ; < 1
INSTRUCTION LIST =
    EXECUTE A TOTAL OF ; 1 RESET W(1)
    EXECUTE A TOTAL OF ; 1 SET W(0)
    EXECUTE A TOTAL OF ; 1 RESET L(0) RESET FLAG
NAME = RN L(0) COUNTER
PRIORITY = 1
INTERRUPTABILITY FLAG = NO
CONCURRENT EXECUTION = NO
ORED PREDECESSOR LIST =
    FLOW CONTROL RN
REQUIRED SEMAPHORE STATUS =
    CHAIN IF ; W(0)
    IS ; SET
    CHAIN IF ; ALT RCV
    IS ; < 1
INSTRUCTION LIST =
    EXECUTE A TOTAL OF ; 1 SET RN L(0)
NAME = RN L(1) COUNTER
PRIORITY = 1
INTERRUPTABILITY FLAG = NO
CONCURRENT EXECUTION = NO
ORED PREDECESSOR LIST =
    FLOW CONTROL RN
REQUIRED SEMAPHORE STATUS =
    CHAIN IF ; W(1)
    IS ; SET
    CHAIN IF ; ALT RCV
    IS ; < 1
INSTRUCTION LIST =
    EXECUTE A TOTAL OF ; 1 SET RN L(1)
NAME = RN L(2) COUNTER
PRIORITY = 1
INTERRUPTABILITY FLAG = NO
CONCURRENT EXECUTION = NO
ORED PREDECESSOR LIST =
    FLOW CONTROL RN
REQUIRED SEMAPHORE STATUS =
    CHAIN IF ; W(2)
    IS ; SET
    CHAIN IF ; ALT RCV
    IS ; < 1
INSTRUCTION LIST =
    EXECUTE A TOTAL OF ; 1 SET RN L(2)
NAME = RESTORE L(2)
PRIORITY = 3
INTERRUPTABILITY FLAG = NO
CONCURRENT EXECUTION = NO
COMPLETED IF RUN CANCELLED = YES

```

```

ORED PREDECESSOR LIST =
  SET ALT RCV
  RESTORE L(2)
REQUIRED SEMAPHORE STATUS =
  CHAIN IF ; FN
  IS ; > 0
  CHAIN IF ; W(2)
  IS ; SET
INSTRUCTION LIST =
  EXECUTE A TOTAL OF ; 1 WRITE DATA L(2)
  EXECUTE A TOTAL OF ; 1 RESET FN
  EXECUTE A TOTAL OF ; 1 RESET SN
  EXECUTE A TOTAL OF ; 1 SET ALTCTR
  EXECUTE A TOTAL OF ; 1 RESET N
ANDED SUCCESSORS =
  CHAIN TO ; RSDONE L(2)
  WITH ITERATIONS THEN CHAIN COUNT OF ;      0
  CHAIN TO ; RESTORE L(2)
  WITH ITERATIONS THEN CHAIN COUNT OF ;      0
NAME = RSDONE L(2)
  PRIORITY =      3
  INTERRUPTABILITY FLAG = NO
  CONCURRENT EXECUTION = NO
  ORED PREDECESSOR LIST =
    RESTORE L(2)
  REQUIRED SEMAPHORE STATUS =
    CHAIN IF ; FN
    IS ; 0
  INSTRUCTION LIST =
    EXECUTE A TOTAL OF ; 1 RESET N
    EXECUTE A TOTAL OF ; 1 RESET RSFLAG
    EXECUTE A TOTAL OF ; 1 RESET FLAG1
    EXECUTE A TOTAL OF ; 1 RESET RN L(2)
NAME = LEVEL 2
  PRIORITY =      0
  INTERRUPTABILITY FLAG = YES
  CONCURRENT EXECUTION = NO
  ORED PREDECESSOR LIST =
    RSDONE L(1)
  INSTRUCTION LIST =
    EXECUTE A TOTAL OF ; 1 RESET W(1)
    EXECUTE A TOTAL OF ; 1 SET W(2)
  ANDED SUCCESSORS =
    CHAIN TO ; PROCESSOR 2
    WITH ITERATIONS THEN CHAIN COUNT OF ;      0
NAME = PROCESSOR 2
  PRIORITY =      0
  INTERRUPTABILITY FLAG = YES
  CONCURRENT EXECUTION = YES
  COMPLETED IF RUN CANCELLED = NO
  STATISTICAL SUCCESSOR STREAM =      5
  ORED PREDECESSOR LIST =
    LEVEL 2
    XMIT GOOD DATA L(2)
    XMIT BAD DATA L(2)
    TIMEOUT DELAY L(2)

```

```

    REXMIT M(2)
REQUIRED SEMAPHORE STATUS =
    CHAIN IF ; FN
    IS ; < 81
    WAIT FOR ; REXMIT WINDOW
    TO BE ; RESET
    WAIT FOR ; ALT RCV
    TO BE ; < 1
    WAIT FOR ; RCVCTR FLAG
    TO BE ; RESET
    WAIT FOR ; RSFLAG
    TO BE ; RESET
    WAIT FOR ; W(2)
    TO BE ; SET
    CHAIN IF ; L2GOOD
    IS ; < 80
    WAIT FOR ; L(1) RESET FLAG
    TO BE ; RESET
STATISTICAL SUCCESSORS =
    CHOOSE AS SUCCESSOR ; 99.97 % XMIT GOOD DATA L(2)
    CHOOSE AS SUCCESSOR ; .03 % XMIT BAD DATA L(2)
INSTRUCTION LIST =
    EXECUTE A TOTAL OF ; 1 RESET A MSG XMIT
NAME = XMIT GOOD DATA L(2)
PRIORITY = 0
INTERRUPTABILITY FLAG = YES
CONCURRENT EXECUTION = YES
ANDED PREDECESSOR LIST =
    PROCESSOR 2
REQUIRED SEMAPHORE STATUS =
    WAIT FOR ; A MSG XMIT
    TO BE ; RESET
INSTRUCTION LIST =
    EXECUTE A TOTAL OF ; 1 READ DATA L(2)
    EXECUTE A TOTAL OF ; 1 SEND GOOD DATA L(2)
    EXECUTE A TOTAL OF ; 1 SET SN
    EXECUTE A TOTAL OF ; 1 SET N
    EXECUTE A TOTAL OF ; 1 SET A MSG XMIT
    EXECUTE A TOTAL OF ; 1 SET L2GOOD
    EXECUTE A TOTAL OF ; 1 SET FN
ANDED SUCCESSORS =
    CHAIN TO ; PROCESSOR 2
    WITH ITERATIONS THEN CHAIN COUNT OF ; 0
    CHAIN TO ; TIMEOUT DELAY L(2)
    WITH ITERATIONS THEN CHAIN COUNT OF ; 0
    CHAIN TO ; RESET LEVEL 1
    WITH ITERATIONS THEN CHAIN COUNT OF ; 0
NAME = XMIT BAD DATA L(2)
PRIORITY = 0
INTERRUPTABILITY FLAG = YES
CONCURRENT EXECUTION = YES
ANDED PREDECESSOR LIST =
    PROCESSOR 2
REQUIRED SEMAPHORE STATUS =
    WAIT FOR ; A MSG XMIT
    TO BE ; RESET

```

```

INSTRUCTION LIST =
  EXECUTE A TOTAL OF ; 1 READ DATA L(2)
  EXECUTE A TOTAL OF ; 1 SEND BAD DATA L(2)
  EXECUTE A TOTAL OF ; 1 SET SN
  EXECUTE A TOTAL OF ; 1 SET N
  EXECUTE A TOTAL OF ; 1 SET RCVCTR FLAG
  EXECUTE A TOTAL OF ; 1 SET A MSG XMIT
  EXECUTE A TOTAL OF ; 1 SET FN
  EXECUTE A TOTAL OF ; 80 RESET L2GOOD
ANDED SUCCESSORS =
  CHAIN TO ; RCV COUNTER
  WITH ITERATIONS THEN CHAIN COUNT OF ; 0
  CHAIN TO ; TIMEOUT DELAY L(2)
  WITH ITERATIONS THEN CHAIN COUNT OF ; 0
  CHAIN TO ; PROCESSOR 2
  WITH ITERATIONS THEN CHAIN COUNT OF ; 0
NAME = TIMEOUT DELAY L(2)
PRIORITY = 0
INTERRUPTABILITY FLAG = YES
CONCURRENT EXECUTION = YES
ORED PREDECESSOR LIST =
  XMIT GOOD DATA L(2)
  XMIT BAD DATA L(2)
REQUIRED SEMAPHORE STATUS =
  CHAIN IF ; FN
  IS ; > 80
INSTRUCTION LIST =
  EXECUTE A TOTAL OF ; 1 TIMEOUT DELAY
ANDED SUCCESSORS =
  CHAIN TO ; REXMIT WINDOW M(2)
  WITH ITERATIONS THEN CHAIN COUNT OF ; 0
  CHAIN TO ; PROCESSOR 2
  WITH ITERATIONS THEN CHAIN COUNT OF ; 0
NAME = RESET LEVEL 1
PRIORITY = 5
INTERRUPTABILITY FLAG = NO
CONCURRENT EXECUTION = NO
ORED PREDECESSOR LIST =
  XMIT GOOD DATA L(2)
REQUIRED SEMAPHORE STATUS =
  CHAIN IF ; L2GOOD
  IS ; > 79
  CHAIN IF ; FN
  IS ; < 81
INSTRUCTION LIST =
  EXECUTE A TOTAL OF ; 1 SET L(1) RESET FLAG
  EXECUTE A TOTAL OF ; 80 RESET L2GOOD
ANDED SUCCESSORS =
  CHAIN TO ; PROCESSOR 1
  WITH ITERATIONS THEN CHAIN COUNT OF ; 0
NAME = REXMIT WINDOW M(2)
PRIORITY = 5
INTERRUPTABILITY FLAG = NO
CONCURRENT EXECUTION = NO
ORED PREDECESSOR LIST =
  TIMEOUT DELAY L(2)

```

```

REQUIRED SEMAPHORE STATUS =
  CHAIN IF ; FN
  IS ; > 80
  WAIT FOR ; W(2)
  TO BE ; SET
INSTRUCTION LIST =
  EXECUTE A TOTAL OF ; 1 SET REXMIT WINDOW
  EXECUTE A TOTAL OF ; 81 SET ALTCTR
  EXECUTE A TOTAL OF ; 1 SET ALT RCV
ANDED SUCCESSORS =
  CHAIN TO ; REXMIT M(2)
  WITH ITERATIONS THEN CHAIN COUNT OF ;      0
NAME = REXMIT M(2)
PRIORITY =      5
INTERRUPTABILITY FLAG = NO
CONCURRENT EXECUTION = NO
ANDED PREDECESSOR LIST =
  REXMIT WINDOW M(2)
REQUIRED SEMAPHORE STATUS =
  WAIT FOR ; REXMIT WINDOW
  TO BE ; SET
INSTRUCTION LIST =
  EXECUTE A TOTAL OF ; 1 WRITE WINDOW L(2)
  EXECUTE A TOTAL OF ; 81 RESET FN
  EXECUTE A TOTAL OF ; 81 RESET SN
  EXECUTE A TOTAL OF ; 81 RESET N
  EXECUTE A TOTAL OF ; 81 RESET PASSCT
  EXECUTE A TOTAL OF ; 81 RESET L2GOOD
  EXECUTE A TOTAL OF ; 1 RESET REXMIT WINDOW
ANDED SUCCESSORS =
  CHAIN TO ; PROCESSOR 2
  WITH ITERATIONS THEN CHAIN COUNT OF ;      0
NAME = RESET ALT RCV L(1)
PRIORITY =      3
INTERRUPTABILITY FLAG = NO
CONCURRENT EXECUTION = NO
ORED PREDECESSOR LIST =
  SET ALT RCV
REQUIRED SEMAPHORE STATUS =
  CHAIN IF ; FN
  IS ; < 1
  CHAIN IF ; W(1)
  IS ; SET
INSTRUCTION LIST =
  EXECUTE A TOTAL OF ; 1 RESET ALT RCV
  EXECUTE A TOTAL OF ; 1 RESET FLAG1
  EXECUTE A TOTAL OF ; 1 RESET RSFLAG
  EXECUTE A TOTAL OF ; 1 RESET RCVCTR FLAG
  EXECUTE A TOTAL OF ; 1 RESET RN L(1)
  EXECUTE A TOTAL OF ; 1 RESET N
NAME = RESET ALT RCV L(2)
PRIORITY =      3
INTERRUPTABILITY FLAG = NO
CONCURRENT EXECUTION = NO
ORED PREDECESSOR LIST =
  SET ALT RCV

```

```

REQUIRED SEMAPHORE STATUS =
  CHAIN IF ; FN
  IS ; < 1
  CHAIN IF ; W(2)
  IS ; SET
INSTRUCTION LIST =
  EXECUTE A TOTAL OF ; 1 RESET ALT RCV
  EXECUTE A TOTAL OF ; 1 RESET FLAG1
  EXECUTE A TOTAL OF ; 1 RESET RSFLAG
  EXECUTE A TOTAL OF ; 1 RESET RCVCTR FLAG
  EXECUTE A TOTAL OF ; 1 RESET RN L(2)
  EXECUTE A TOTAL OF ; 1 RESET N
NAME = RESET LEVEL L(1)
PRIORITY = 1
INTERRUPTABILITY FLAG = NO
CONCURRENT EXECUTION = NO
ORDER PREDECESSOR LIST =
  FLOW CONTROL RN
REQUIRED SEMAPHORE STATUS =
  CHAIN IF ; L(1) RESET FLAG
  IS ; SET
  CHAIN IF ; FN
  IS ; < 1
INSTRUCTION LIST =
  EXECUTE A TOTAL OF ; 1 RESET W(2)
  EXECUTE A TOTAL OF ; 1 SET W(1)
  EXECUTE A TOTAL OF ; 1 RESET L(1) RESET FLAG

```

```

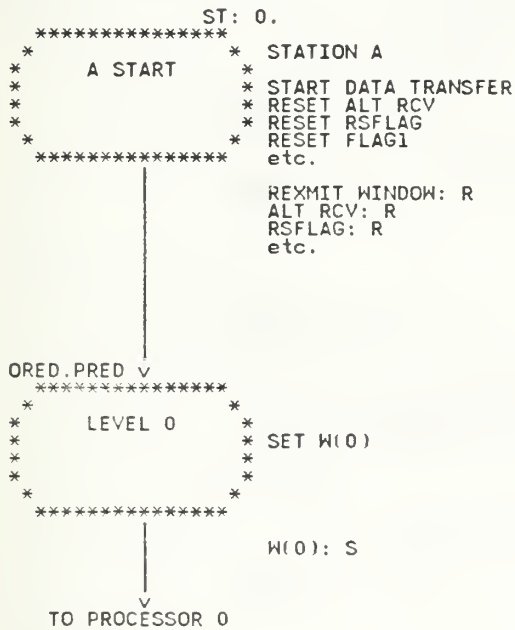
***** FILES - SYS.FILE.SET
SOFTWARE TYPE = FILE
NAME = GENERAL STORAGE
NUMBER OF BITS = 2000000.
INITIAL RESIDENCY =
  A MEMORY
READ ONLY FLAG = NO

```

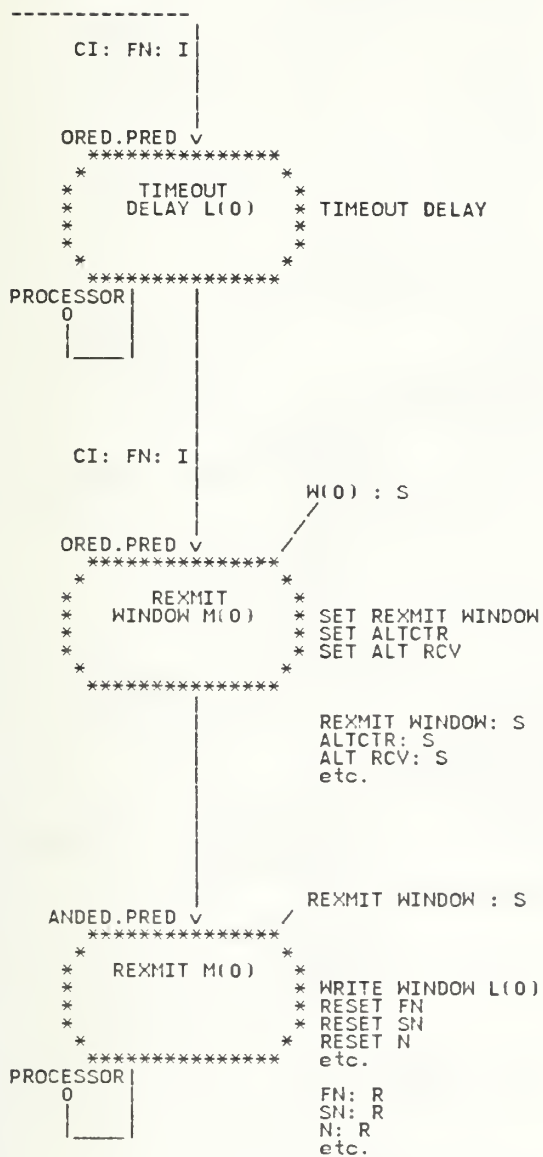
APPENDIX B. SIMULATION FLOW DIAGRAM

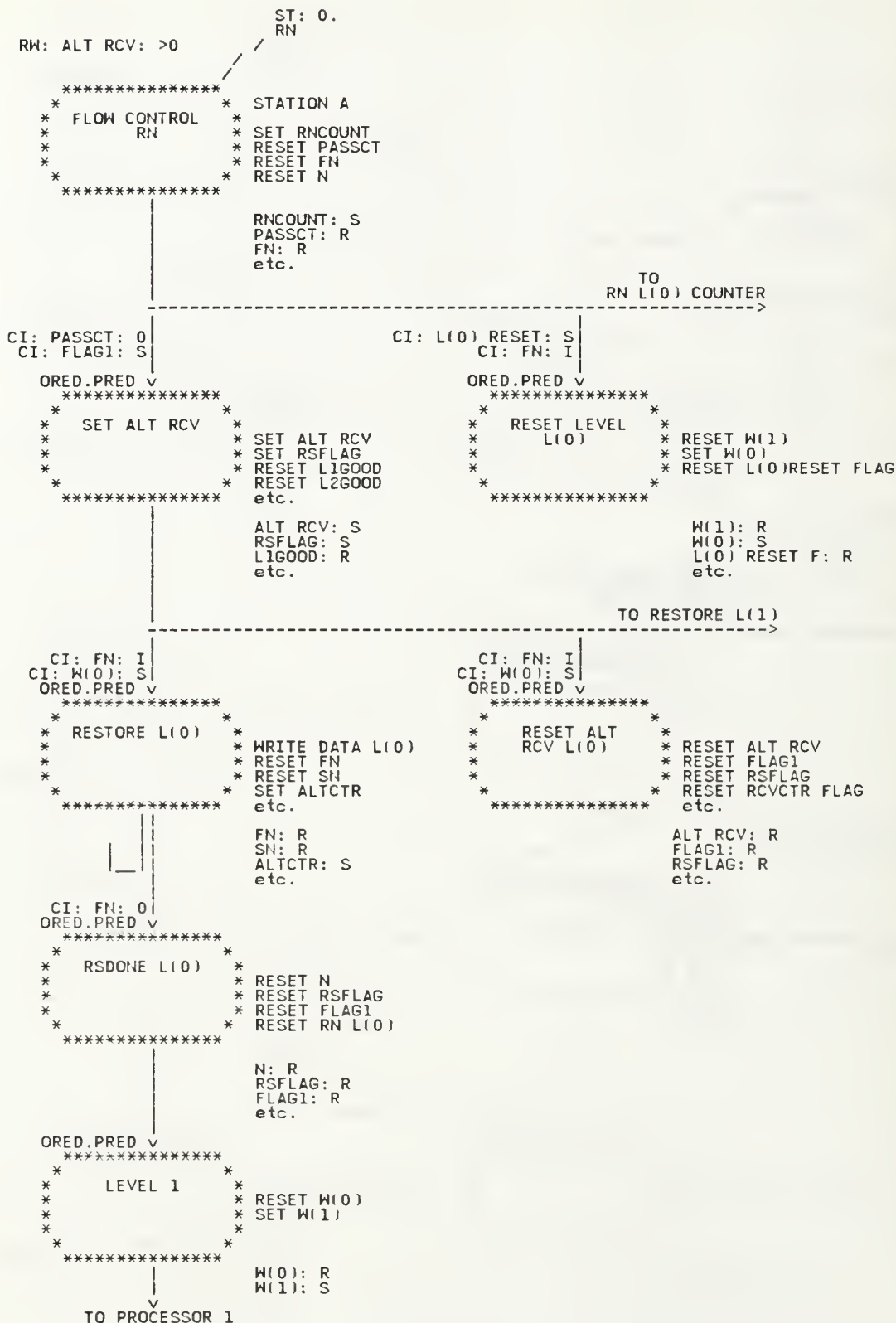
CACI NETIN RELEASE 4.01

Adaptive GBN with three stages







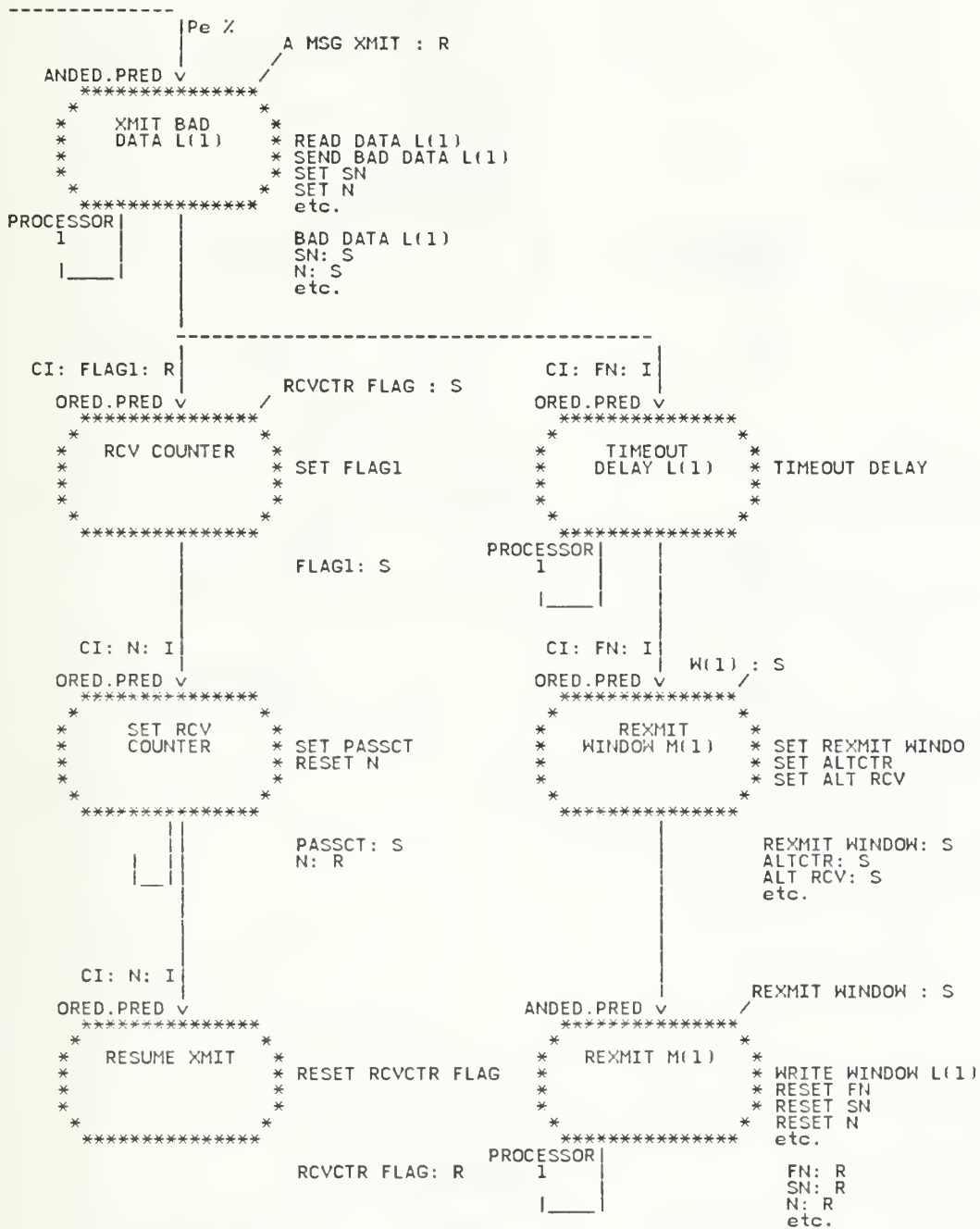




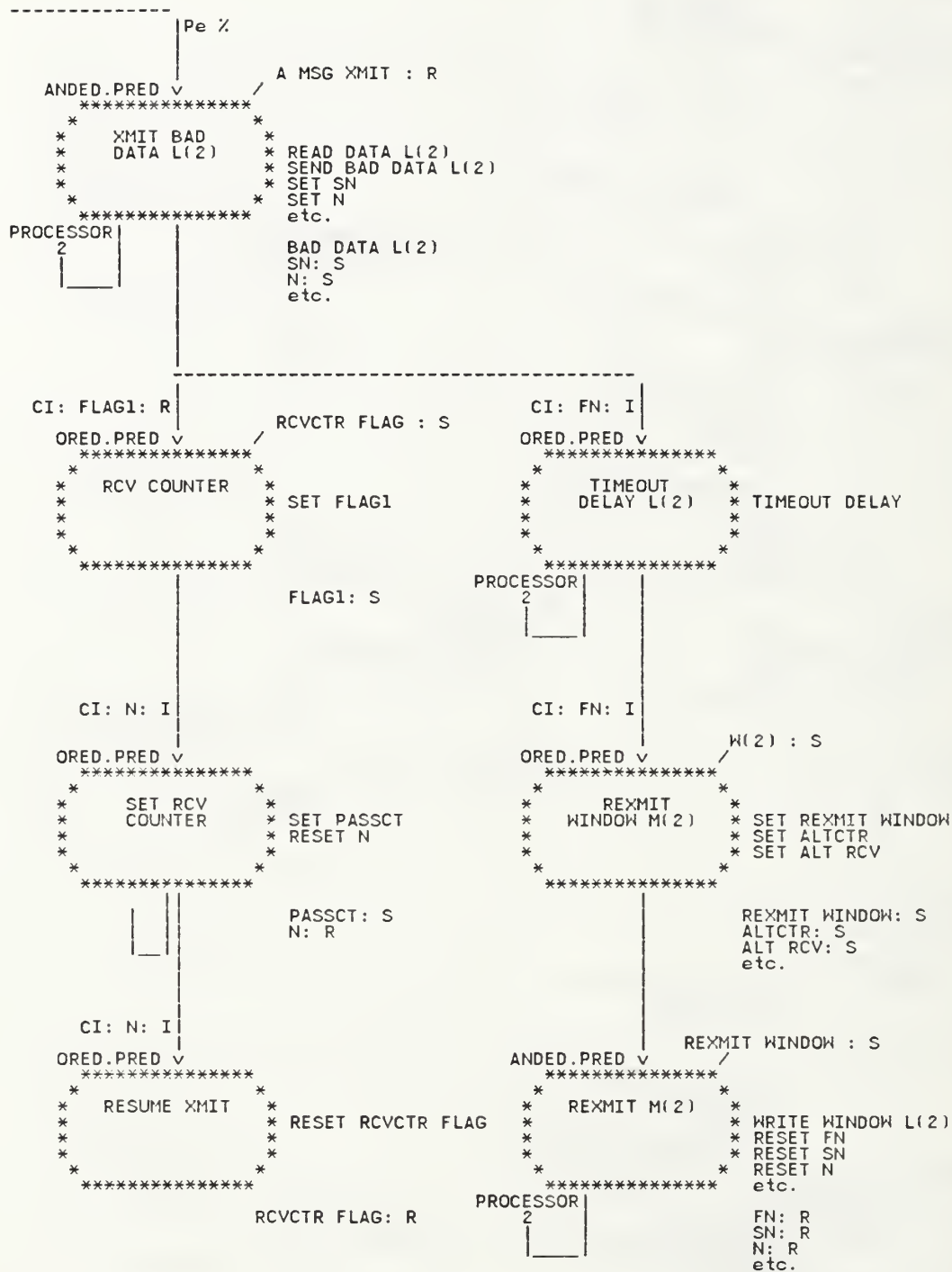
TO RN L(2) COUNTER
>

<p>CI: W(0): S CI: ALT RCV: I</p> <p>ORED.PRED ↓</p> <pre> ***** * * * RN L(0) * * COUNTER * * * * * * * ***** </pre> <p style="text-align: right;">* SET RN L(0)</p> <p style="text-align: center;">RN L(0): S</p>	<p>CI: W(1): S CI: ALT RCV: I</p> <p>ORED.PRED ↓</p> <pre> ***** * * * RN L(1) * * COUNTER * * * * * * * ***** </pre> <p style="text-align: right;">* SET RN L(1)</p> <p style="text-align: center;">RN L(1): S</p>
---	---

<p>CI: W(2): S CI: ALT RCV: I</p> <p>ORED.PRED ↓</p> <pre> ***** * * * RN L(2) * * COUNTER * * * * * * * ***** </pre> <p style="text-align: right;">* SET RN L(2)</p> <p style="text-align: center;">RN L(2): S</p>	<p>CI: L(1) RESET: S CI: FN: I</p> <p>ORED.PRED ↓</p> <pre> ***** * * * RESET LEVEL * * L(1) * * * * * * * ***** </pre> <p style="text-align: right;"> * RESET W(2) * SET W(1) * RESET L(1) RESET FLAG </p> <p style="text-align: center;"> W(2): R W(1): S L(1) RESET FLAG: R etc. </p>
---	--



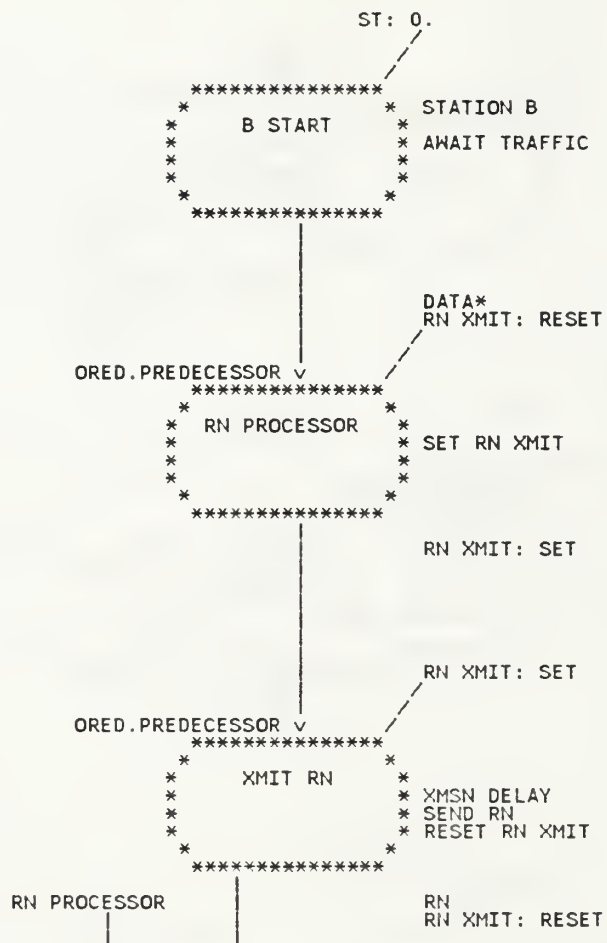




```

ST: 0.
RN
ALT RCV: > 0
/
RW: ALT RCV: > 0
*****
*          *
*  ALT FLOW  *
*  CONTROL  *
*          *
*          *
*          *
*****
STATION A
RESET ALTCTR
*****
ALTCTR: RESET
*****
CI: ALTCTR: < 1
*****
ORED.PREDECESSOR ✓
*****
*          *
*  ALT 0    *
*          *
*          *
*          *
*****
RESET ALT RCV
RESET RCVCTR FLAG
RESET N
RESET FLAG1
*****
ALT RCV: RESET
RCVCTR FLAG: RESET
N: RESET
etc.

```



LIST OF REFERENCES

1. K.M. Sundara Murthy, et al., "VSAT User Network Examples", *IEEE Communications*, Vol. 27, No. 5, May 1989, pp. 50-57.
2. Mark Maggenti, Tri T. Ha, and Timothy Pratt, "VSAT Networks-An Overview", *International Journal of Satellite Communications*, Vol. 5, pp. 219-225, Wiley and Sons, Ltd., March 1987.
3. Tri T. Ha, "Personal Computer Communications via VSAT Networks", *IEEE Journal on Selected Areas in Communications*, Vol. 7, No. 2, pp. 235-245, February 1987.
4. Rodger E. Ziemer and Roger L. Peterson, *Digital Communications and Spread Spectrum Systems*, Macmillan Publishing Co., New York, New York, 1985.
5. David P. Hayes, *Performance Analysis of VSAT Networks*, Master's thesis, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, March 1987.
6. Mark A. Maggenti, *Spread Spectrum Satellite Multiple Access and Overlay Service*, Master's thesis, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, March 1987.
7. Dimitri Bertsekas and Robert Gallager, *Data Networks*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1987.
8. Paul E. Green, editor, *Computer Network Architectures and Protocols*, Plenum Press, Yorktown Heights, New York, 1982.
9. Y. Ishibashi and A. Iwabuchi, "Throughput Analysis of adaptive ARQ Schemes", *Institute of Electronics and Communications Engineers of Japan*, Vol. 71B, No. 6, pp. 698-707, June 1988.

10. Kurtis J. Guth, "*An Adaptive ARQ Strategy for Packet Switching Data Communication Networks*", Master's thesis, Naval Postgraduate School, Monterey, California, 1989.
11. NETWORK II.5, Version 4.01, CACI Products Company, La Jolla, California, October 1988.

INITIAL DISTRIBUTION LIST

	No. Copies
1. Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2
2. Library, Code 0142 Naval Postgraduate School Monterey, CA 93943-5002	2
3. Chairman, Code 62 Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, CA 93943-5000	1
4. Professor Tri T. Ha, Code 62Ha Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, CA 93943-5000	5
5. Professor Glen A. Meyers, Code 62Mv Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, CA 93943-5000	1
6. CPT David T. Tsuda DCA-Guam Field Office NAVCAMS WLSTPAC, Box 141 FPO San Francisco 96630-1837	2
7. Commander Defense Communications Agency ATTN: B400 Washington, D.C. 20305	2

JUN 12 1995

OCT 26 1995

MAR -4 1996

3

GAYLORD 83

Keep this card in the book pocket
Book is due on the latest date stamped
on the latest date stamped

Thesis

T02217 Tsuda

c.1 Adaptive Go-Back-N:an
ARQ protocol for a tacti-
cal VSAT network.



thesT82317

Adaptive Go-Back-N :



3 2768 000 86743 6

DUDLEY KNOX LIBRARY